



# Map All The Things

Geolocation for Mobile Forensics Practitioners

Kim Thomson  
H-11 Digital Forensics  
[kim@h11dfs.com](mailto: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](mailto:kim@h11dfs.com)

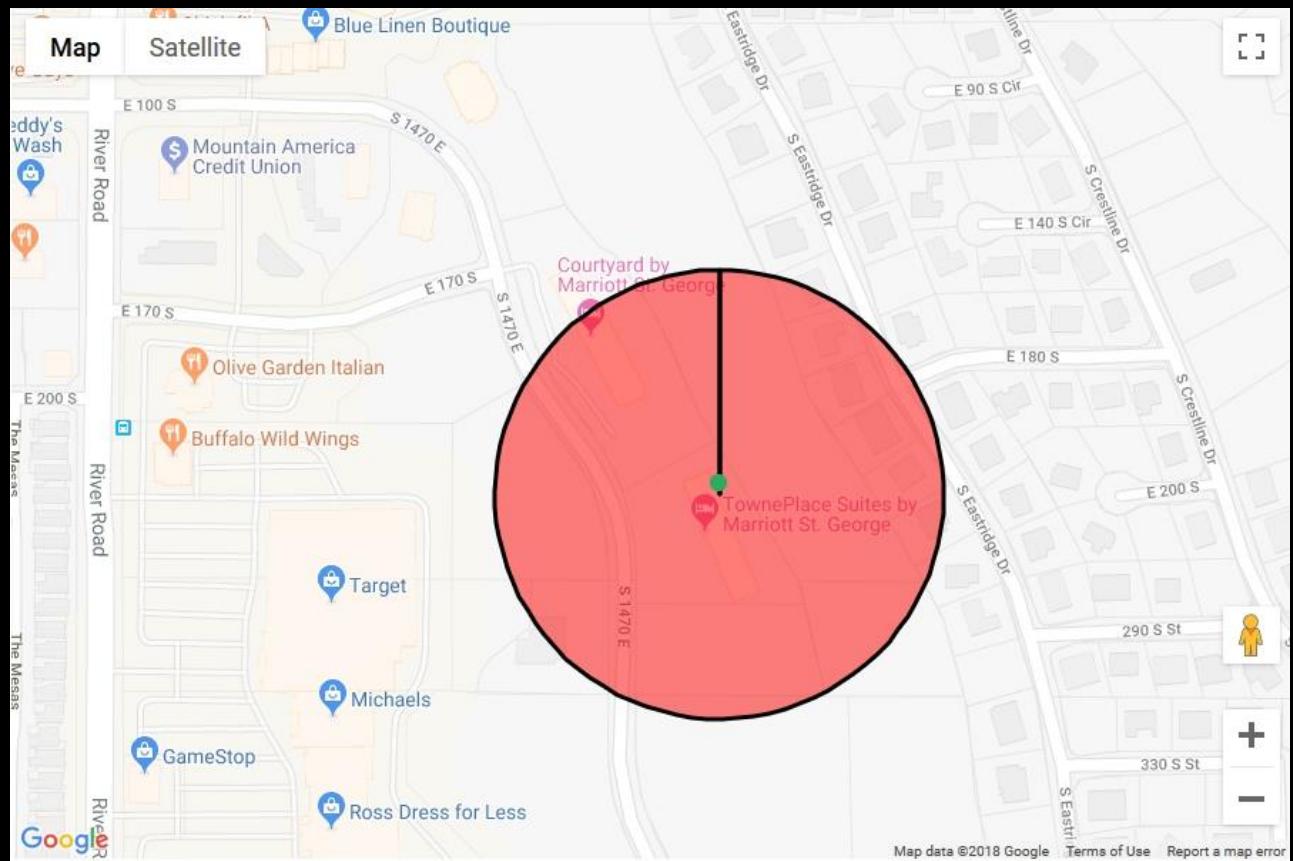
**geo-location = what + when + where**

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**

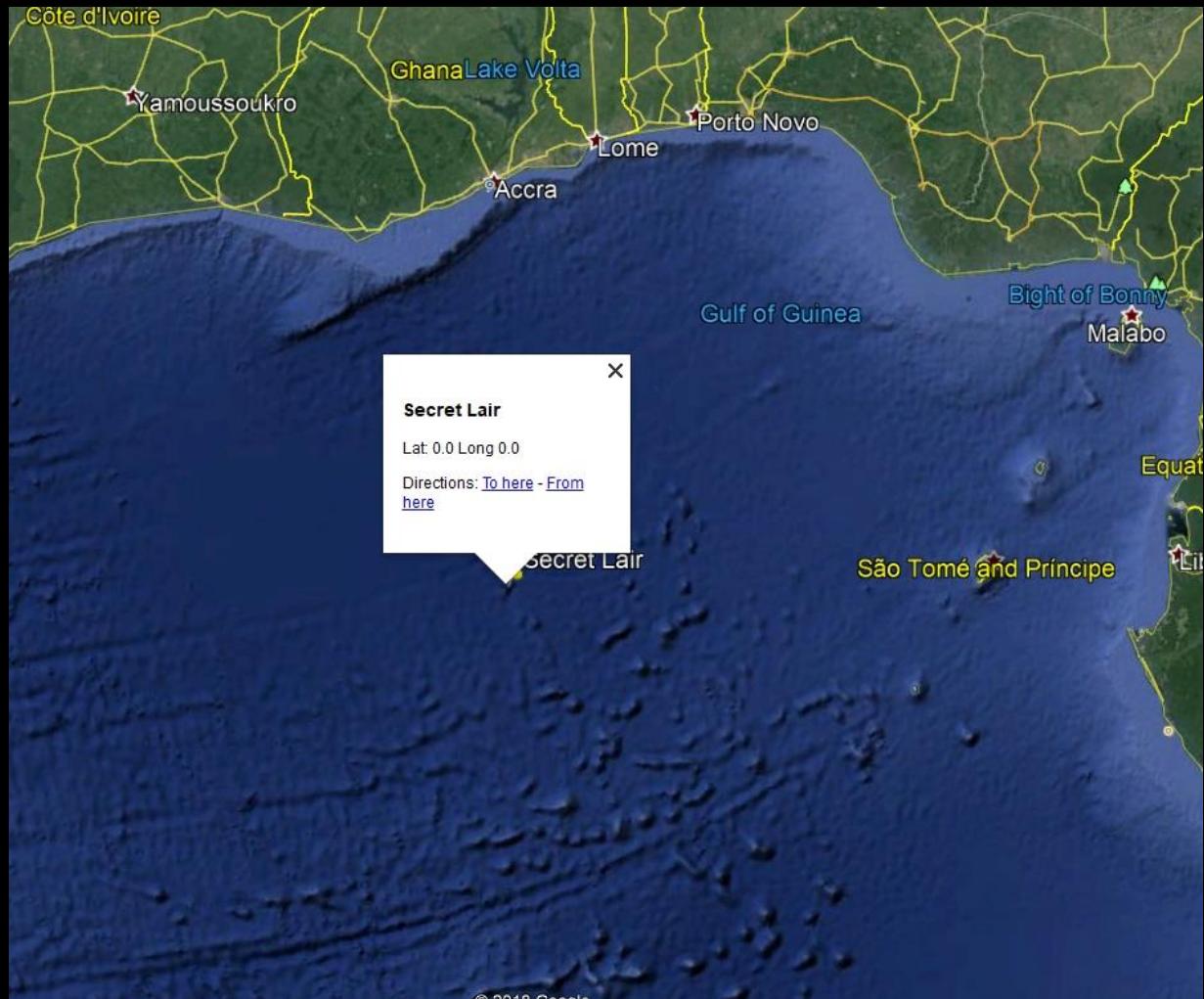
# what can we do with it?

- 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 or 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

The image shows a terminal window and a file explorer side-by-side.

The terminal window displays the output of the command "dumpsys". It includes a table of latest scan results for WiFi networks and a list of generated log files:

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				
8a:f2:9a:99:a1:20				

The list of files generated by dumpsys:

- dumpsystxt
- package.txt
- wifi.txt
- netstats.txt
- dbinfo.txt
- bluetooth\_manager.txt
- notification.txt
- usagestats.txt
- storaged.txt
- batterystats.txt
- activity.txt
- jobscheduler.txt
- nfc.txt
- content.txt
- graphicsstats.txt
- CustomFrequencyManagerService.txt
- shortcut.txt
- procstats.txt
- appops.txt
- media.camera.txt
- settings.txt
- wifiscanner.txt

The file explorer shows a folder named "dumpsys\_files" containing the same list of files as the terminal output.

# 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) – 1<sup>st</sup> three bytes, identifies vendor/manufacturer
  - Last three bytes – specific device

WiGLE WiFi		
Run: 26	0	Lat: 13.69368282 +/- 106 ft
0	0	Lon: -89.23754352 Alt: 2,718 ft
0	0	Speed: 0 mph Sats: 6
20 scanned in 2837ms. DB Queue: 0		
-61	barcelo	Ruckus Wireless - 13:54:00
-69	barcelo	Ruckus Wireless - 13:54:00
-71	barcelo	Ruckus Wireless - 13:54:00
-75	barcelo	Ruckus Wireless - 13:54:00
-76	barcelo	Ruckus Wireless - 13:54:00
-78	barcelo	Ruckus Wireless - 13:54:00
-78	barcelo	Ruckus Wireless - 13:54:00
-83	barcelo	Ruckus Wireless - 13:54:03
-84	barcelo	Ruckus Wireless - 13:54:00

# 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

Web History Go to ▾

Title:	SLCC Web Authentication
Last Visited:	2016-01-20 08:12:52(UTC-7)
Visits:	
URL:	<a href="https://wism.slcc.edu/fs/customwebauth/login.html?switch_url=https://wism.slcc.edu/login.html&amp;ap_mac=88:75:56:c7:7f:30&amp;client_mac=a0:39:f7:9d:00:&amp;wlan=lh-open&amp;redirect=www.google.com/">https://wism.slcc.edu/fs/customwebauth/login.html? switch_url=https://wism.slcc.edu/ login.html&amp;ap_mac=88:75:56:c7:7f:30&amp;client_mac=a0: 39:f7:9d:00:&amp;wlan=lh- open&amp;redirect=www.google.com/</a>
Source:	Chrome : synced data: LG-H811

# 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

← WiGLE WiFi – Network :

 T-Mobile USA

**310260\_11622\_24775171**

Signal: -101 Type: GSM First Seen: 12:51:29  
Capabilities: IWLAN;us  
Channel: IWLAN ARFCN Observations:(querying...)  
2300  
Status: Operational Brand: T-Mobile  
Frequencies: GSM 1900 / UMTS 1900 / UMTS 1700 /  
LTE 850 / LTE 700 / LTE 1900 / LTE 1700  
Notes: Former Cook Inlet West Wireless, Voicestream;  
now universal USA code. Also used for Ting.

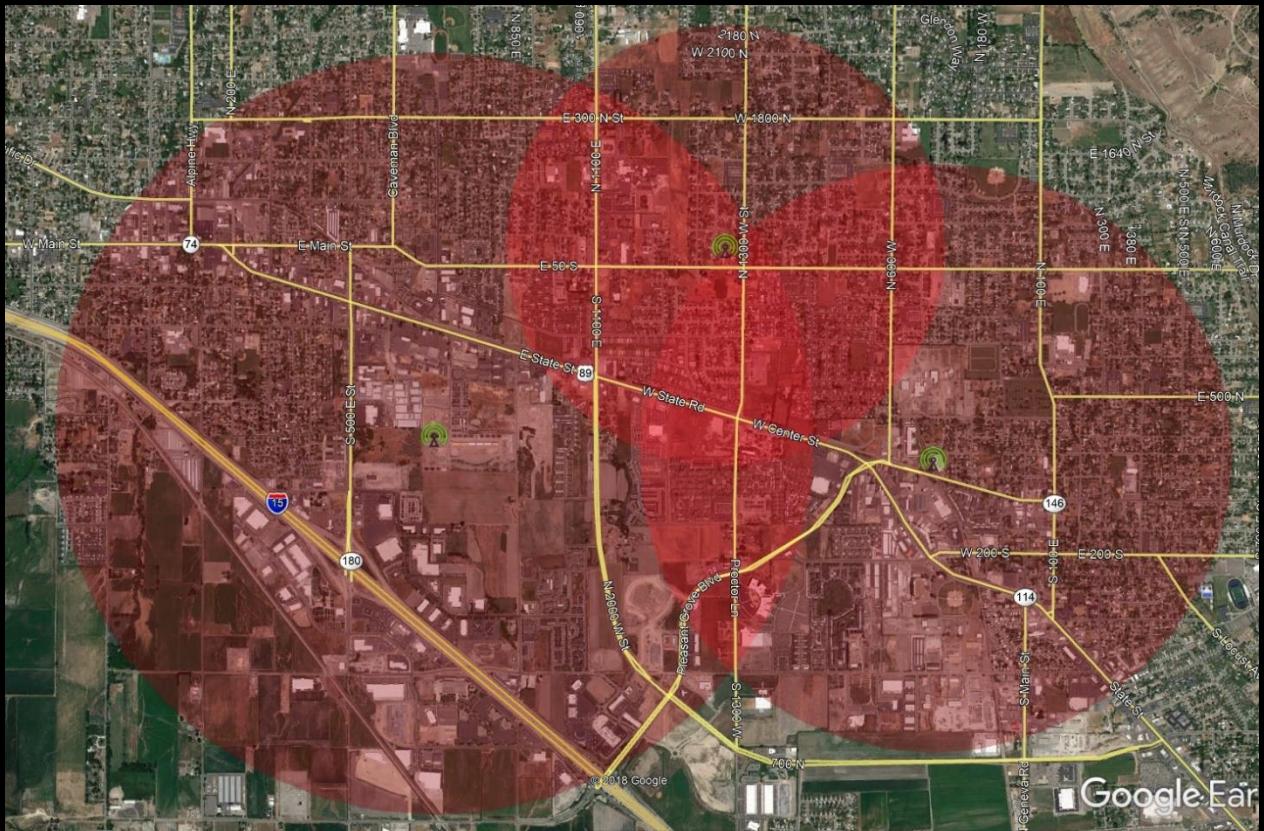
# cell towers - connected vs. seen

## 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)

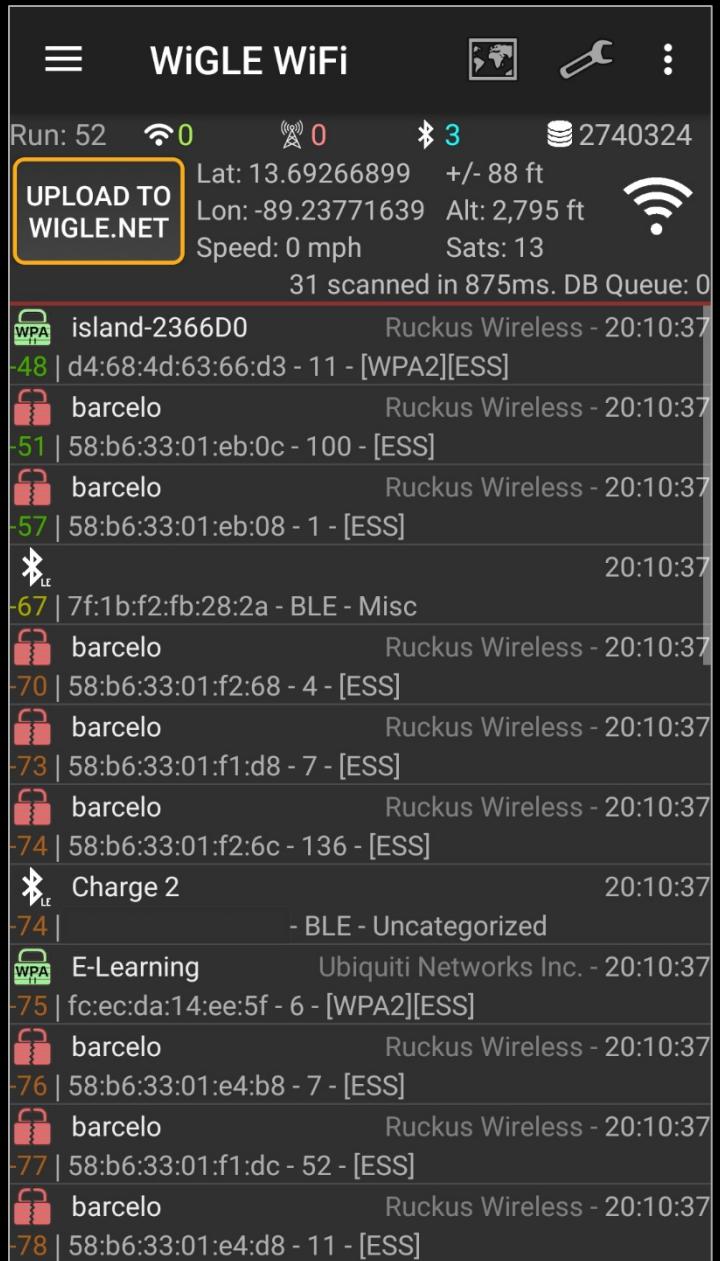
lat = location_result[u'location'][u'lat']
lng = location_result[u'location'][u'lng']
accuracy = location_result[u'accuracy']
address = gmaps.reverse_geocode((lat, lng))[0][u'formatted_address']

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

Google Geolocation/Geocoding API – Python

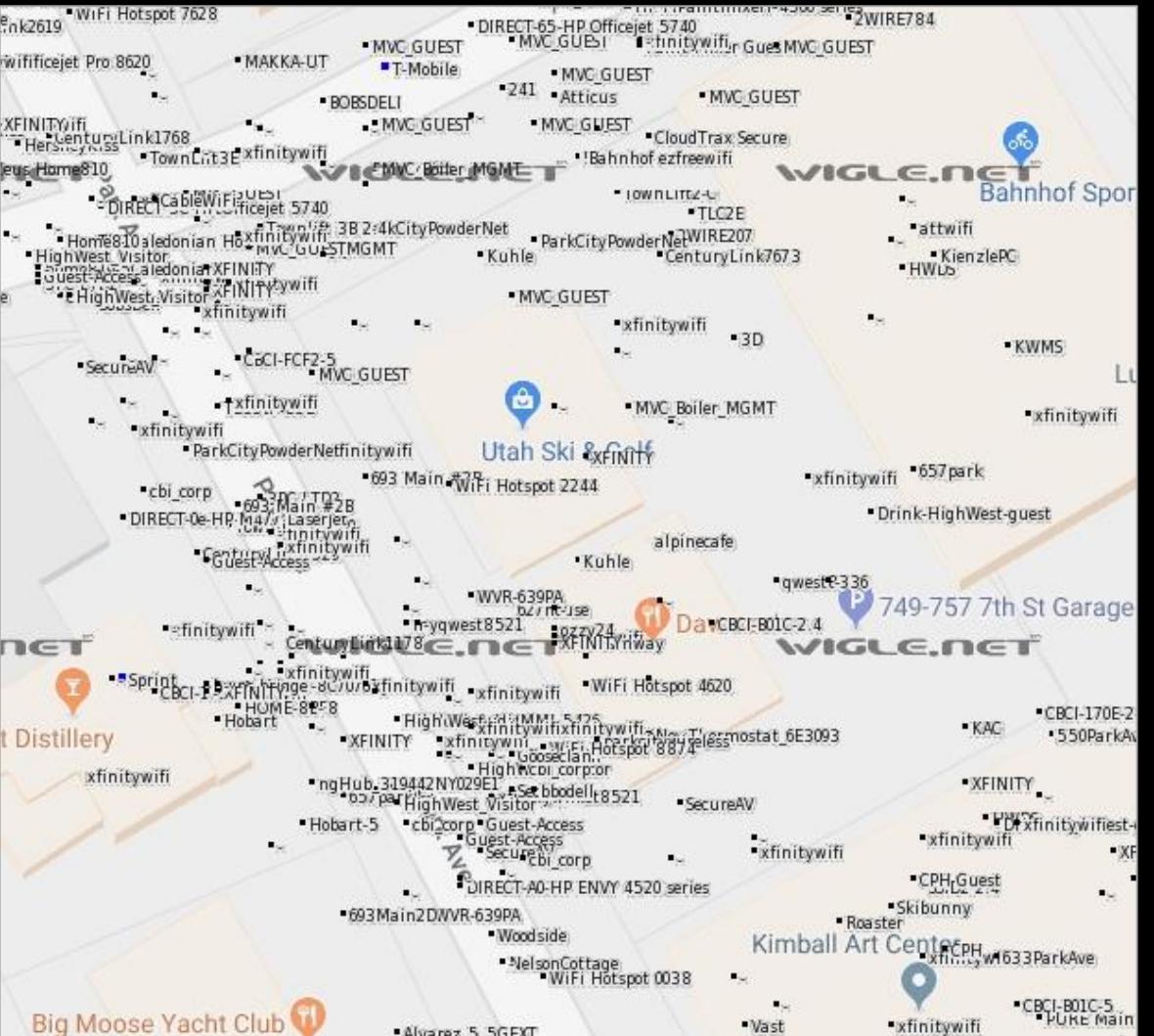
# wigle.net

- 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](mailto: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](mailto: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"
```

## 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",  
            "lastupd": "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,  
            "lat": 33.45885086,  
            "lon": -111.98564911  
        }  
    ]  
}
```

## Request URL

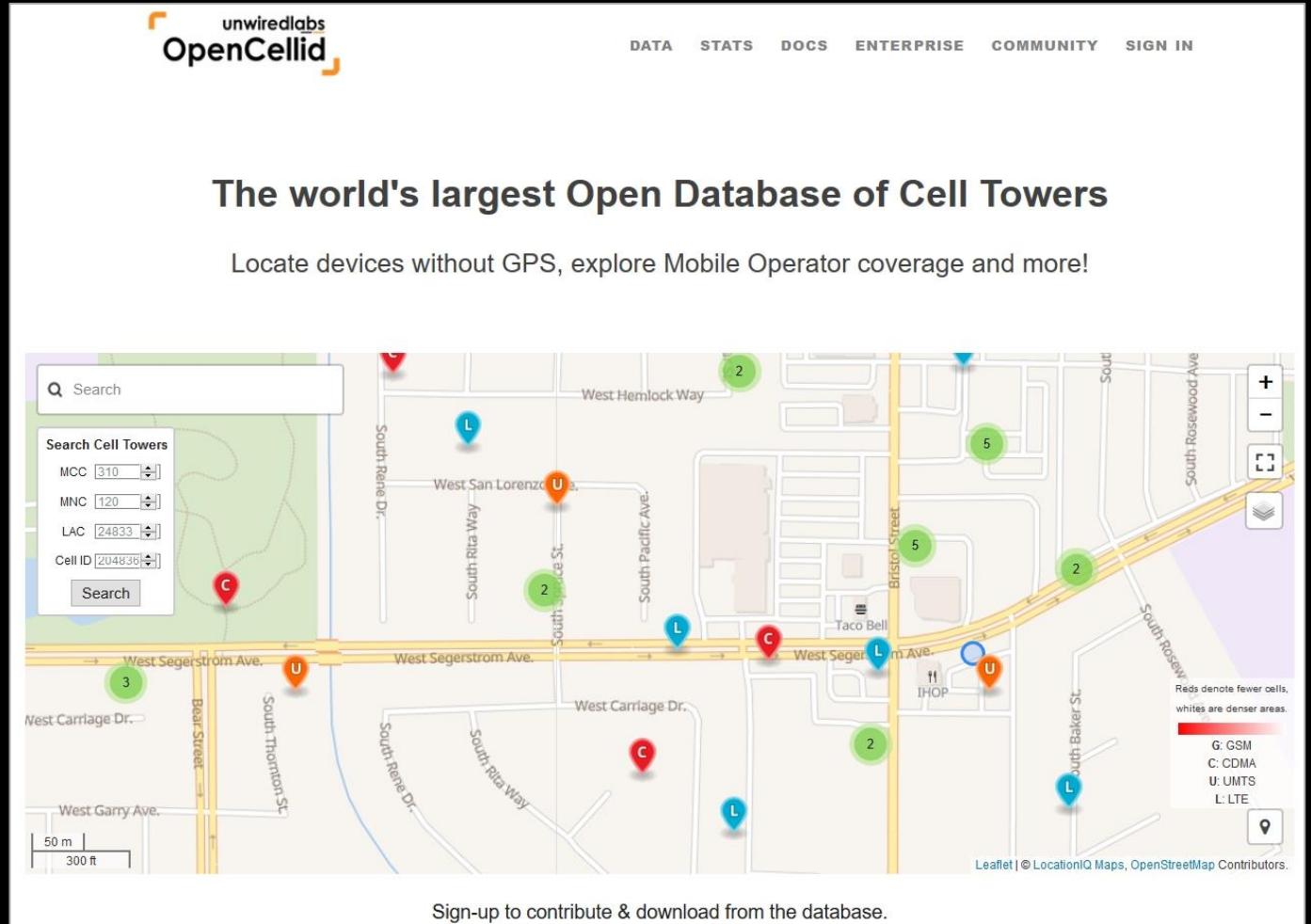
<https://api.wigle.net/api/v2/network/search?onlymine=false&first=0&freenet=1>

[https://api.wigle.net/swagger#/Network search and information tools/](https://api.wigle.net/swagger#/Network%20search%20and%20information%20tools/)

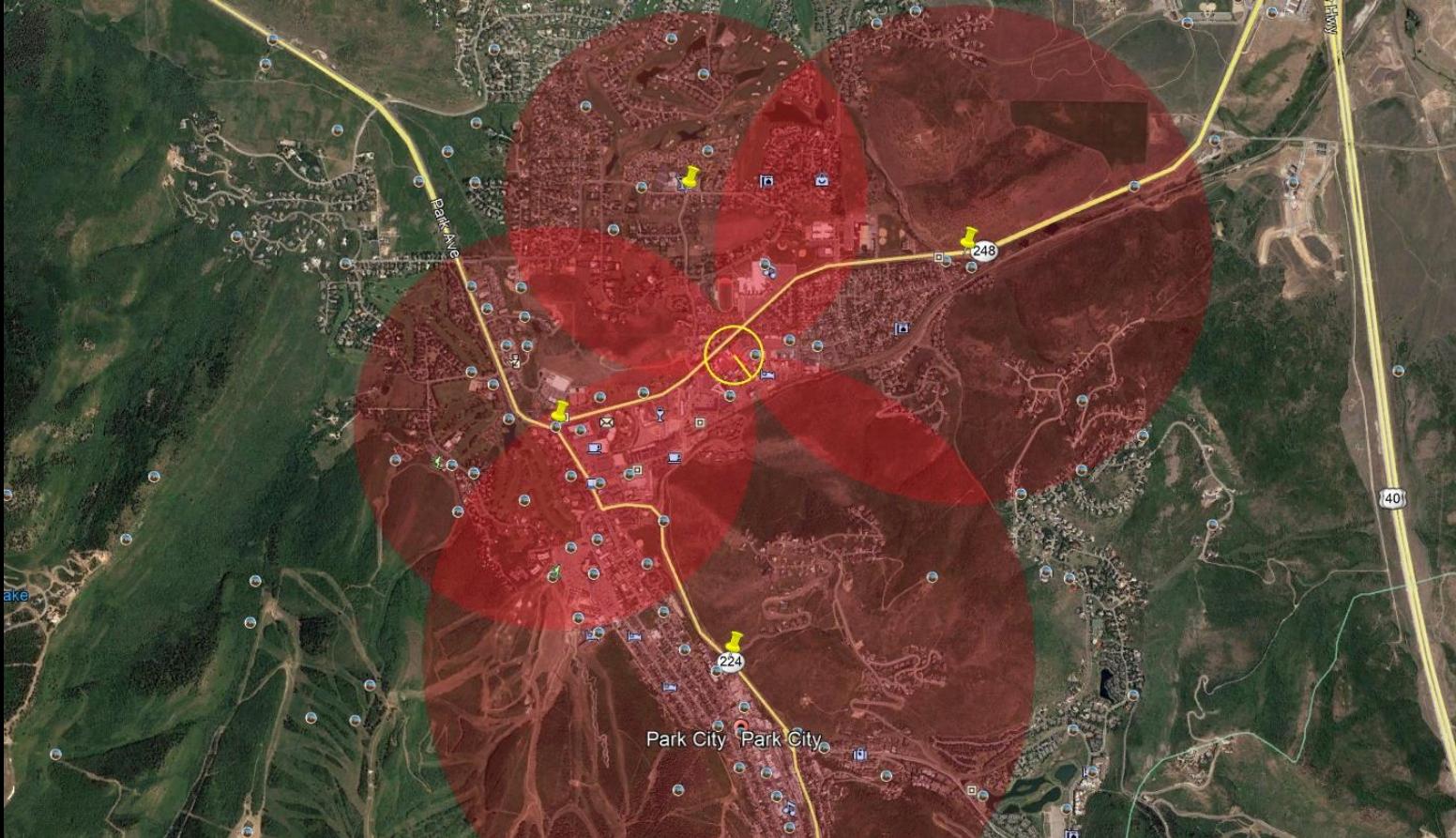
# opencellid.org

- 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>



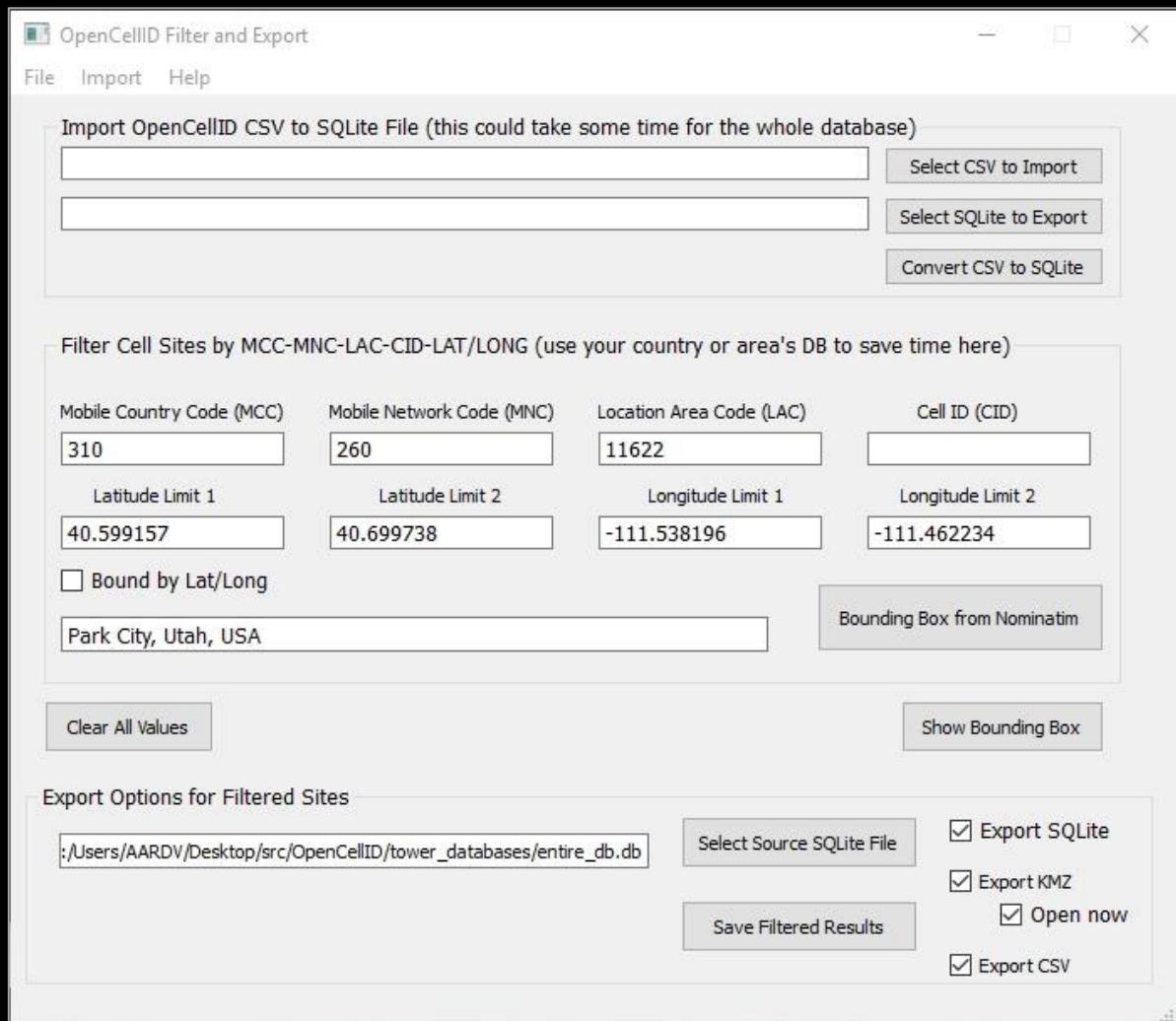
# weighted-centroid trilateration



Multiple Measurements with differing signal strengths

# opencellid.org

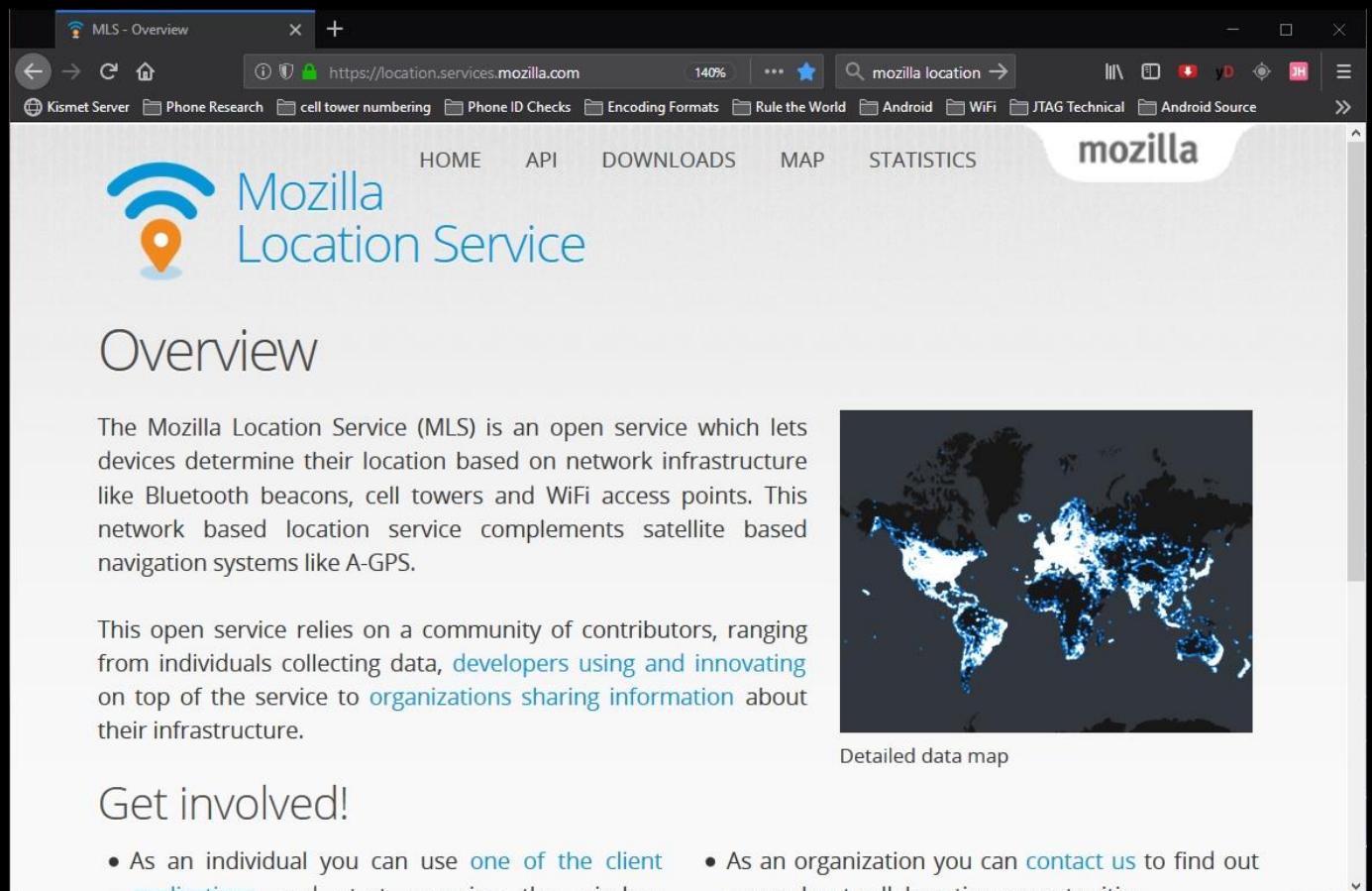
- OpenCellID Filter and Export
- Python2.7 and QT4 (yeah, yeah, I know)
- Currently rewriting in Python3 and QT5



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

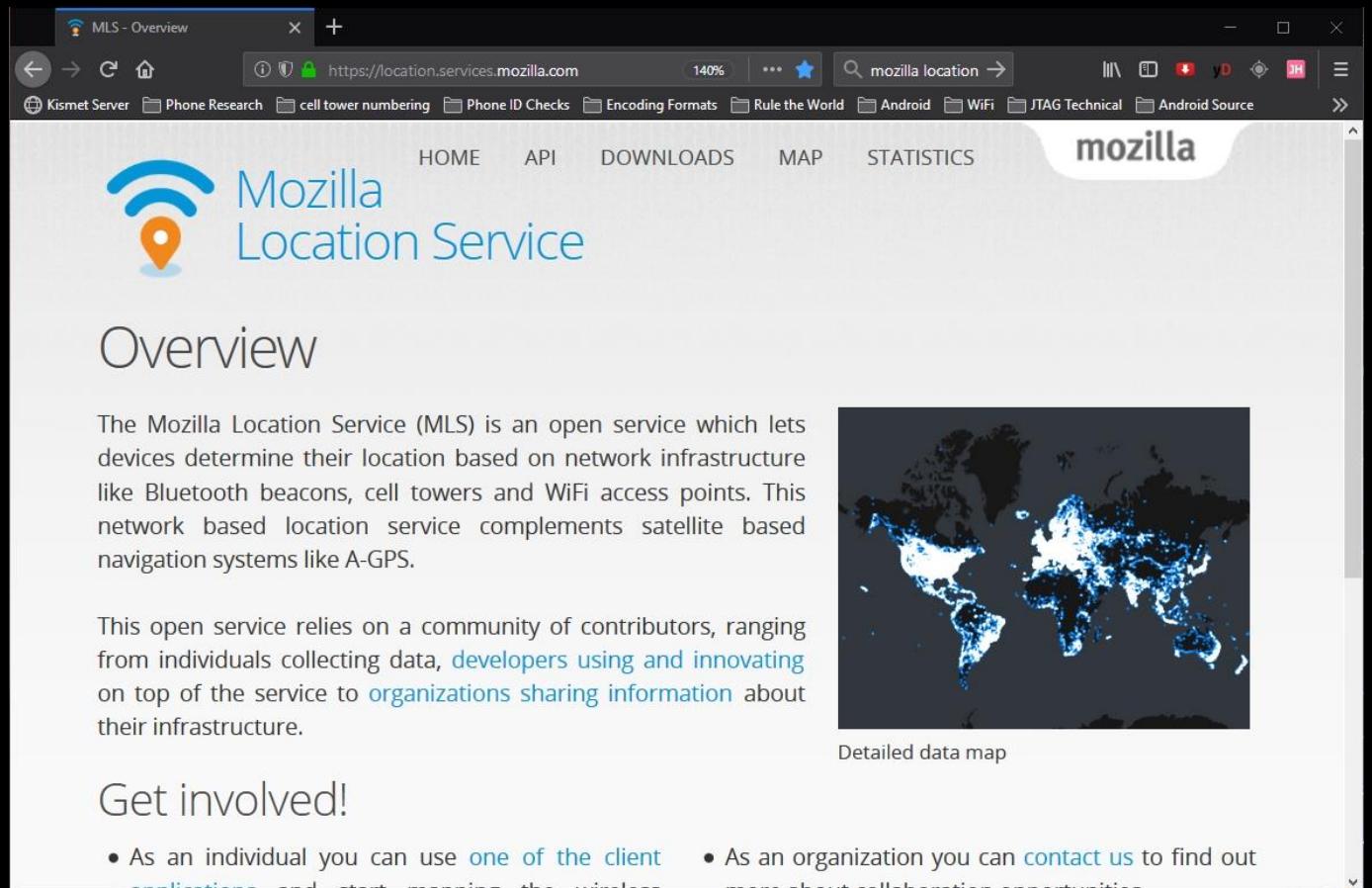


The screenshot shows a web browser window titled "MLS - Overview". The address bar displays the URL <https://location.services.mozilla.com>. The page content is the "Overview" section of the Mozilla Location Service. It features a logo with a blue Wi-Fi icon and an orange location pin. The text explains that the Mozilla Location Service (MLS) is an open service that uses network infrastructure like Bluetooth beacons, cell towers, and WiFi access points to determine device location. It complements satellite-based navigation systems like A-GPS. Below this text is a map of the world with glowing blue dots representing data collection points, primarily concentrated in North America, Europe, and Asia. A caption below the map reads "Detailed data map". At the bottom, there are two bullet points under the heading "Get involved!":

- As an individual you can use [one of the client applications](#) and start mapping the wireless...
- As an organization you can [contact us](#) to find out more about collaboration opportunities...

# 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

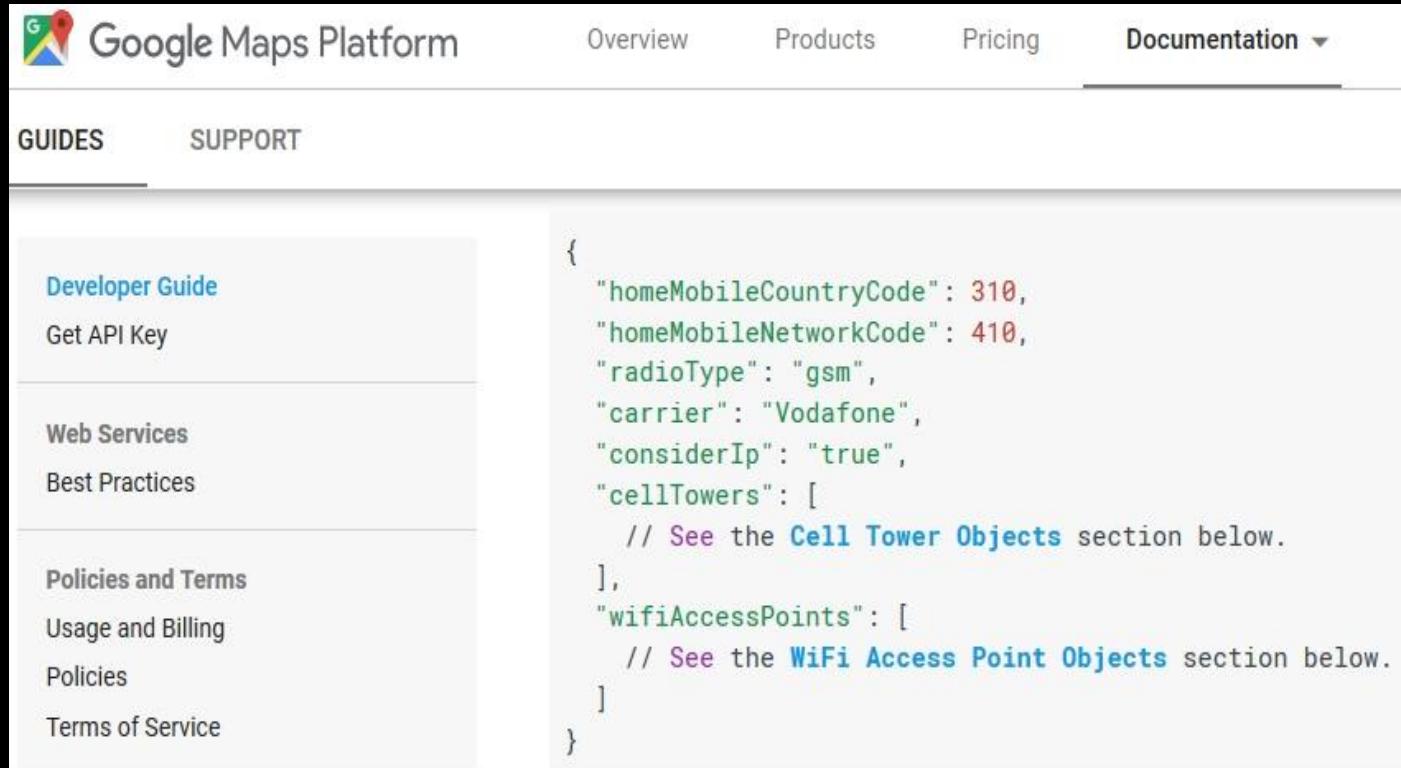


The screenshot shows a web browser window titled "MLS - Overview". The address bar displays the URL <https://location.services.mozilla.com>. The page content is the "Overview" section of the Mozilla Location Service. It features a logo with a blue Wi-Fi icon and an orange location pin. The text describes the service as an open network-based location service that complements satellite-based navigation systems like A-GPS. It mentions that the service relies on a community of contributors, including individuals collecting data and organizations sharing information. Below this text is a "Detailed data map" showing a world map with glowing blue and white points of data density. At the bottom, there are two bullet points under the heading "Get involved!":

- As an individual you can use [one of the client applications](#) and start mapping the wireless...
- As an organization you can [contact us](#) to find out more about collaboration opportunities...

# 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)



The screenshot shows the Google Maps Platform Developer Guide page. The left sidebar contains links for 'Developer Guide' (which is active), 'Get API Key', 'Web Services', 'Best Practices', 'Policies and Terms', 'Usage and Billing', 'Policies', and 'Terms of Service'. The main content area displays a JSON object representing location data:

```
{  
  "homeMobileCountryCode": 310,  
  "homeMobileNetworkCode": 410,  
  "radioType": "gsm",  
  "carrier": "Vodafone",  
  "considerIp": "true",  
  "cellTowers": [  
    // See the Cell Tower Objects section below.  
  ],  
  "wifiAccessPoints": [  
    // See the WiFi Access Point Objects section below.  
  ]  
}
```

# 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

```
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

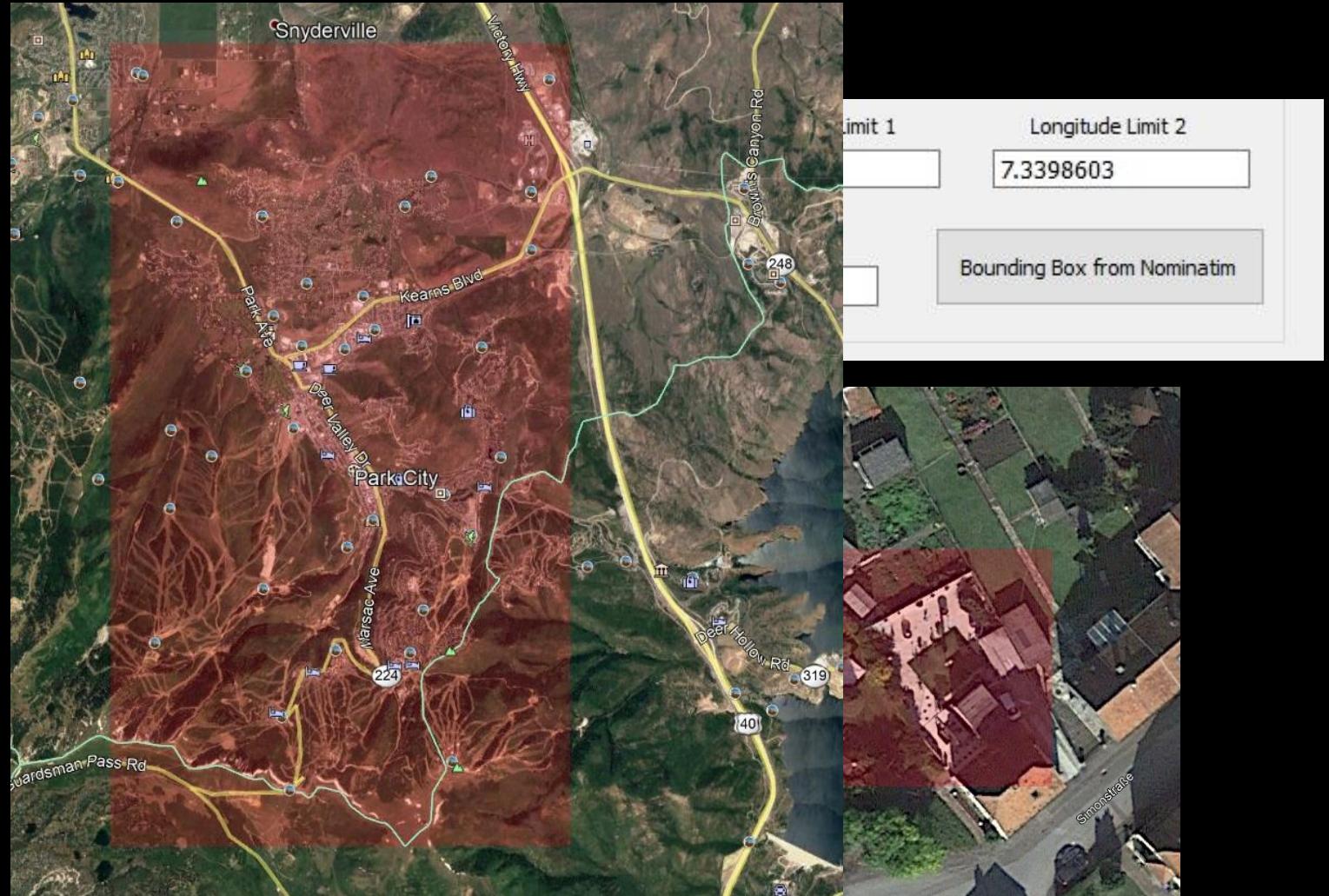
The screenshot shows the OpenStreetMap website interface. At the top, there is a search bar with the placeholder "Search" and a "Go" button. To the right of the search bar are buttons for "Edit", "History", and "Export". On the far right, there are links for "GPS Traces" and "User Diaries". Below the search bar, a modal window is open for "Way: Sidewinder Drive (10162099)". The modal title is "Way: Sidewinder Drive (10162099)". Inside, it says "#maproulette Help\_fix\_old\_TIGER\_ways! Park City Details". It was edited about 1 year ago by torapa, Version #11 · Changeset #50612331. A "Tags" section lists the following key-value pairs:

highway	residential
name	Sidewinder Drive
tiger:cfcc	A41
tiger:county	Summit, UT
tiger:name_base	Sidewinder
tiger:name_type	Dr

The main map view shows a residential area with several streets labeled, including Kearns Boulevard, Cooke Drive, Little Bessie Avenue, Comstock Drive, Ina Avenue, Monarch Drive, Gold Dust Lane, and Berrett Lane. A red line highlights the path of Sidewinder Drive. Other features include a school building labeled "Park City High School" and various parks and trails.

# OpenStreetMap - Nominatim

- geocoding
- named location searching
- bounding boxes for filtering

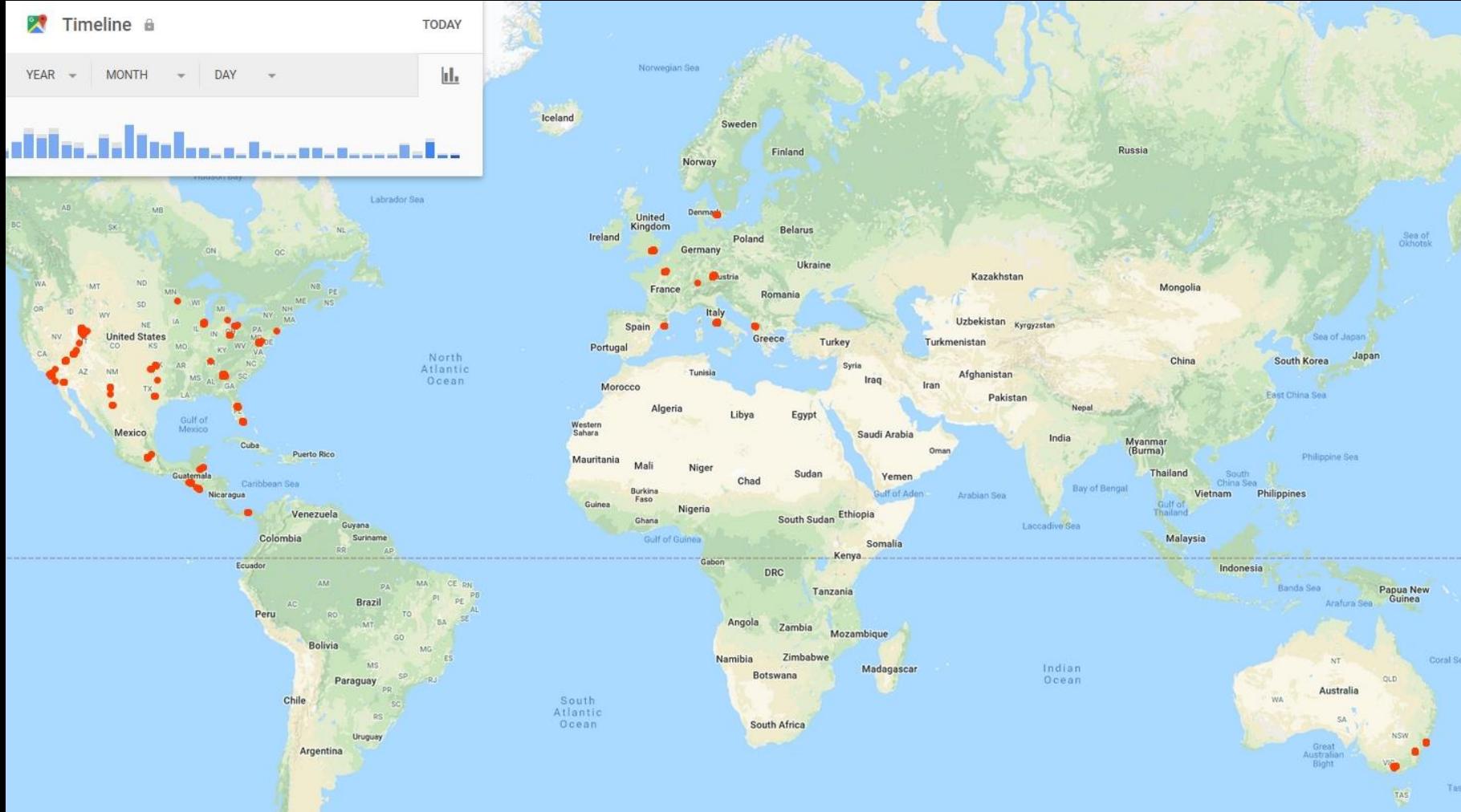


# skyhook

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

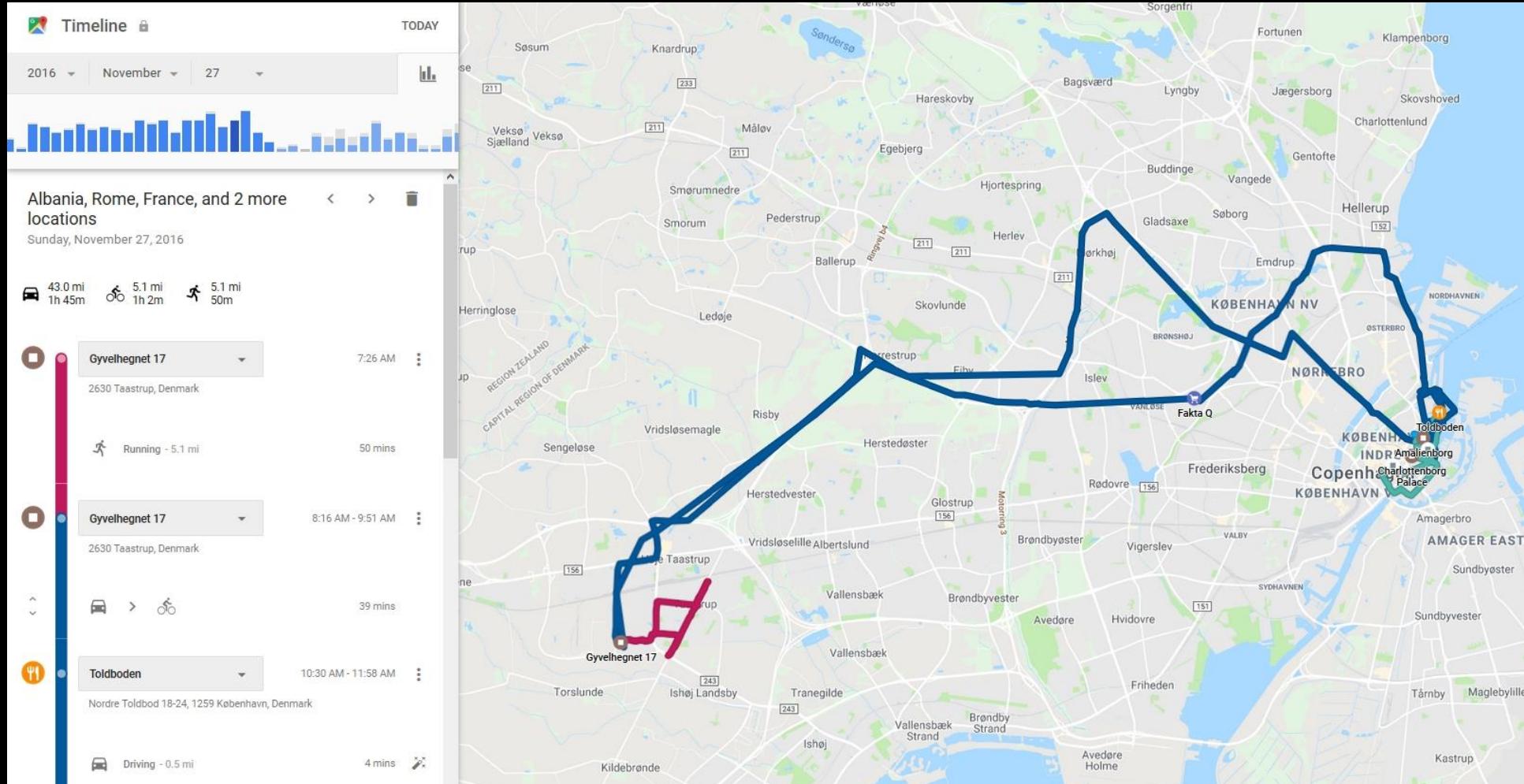


# google location history



Overall, showing total history

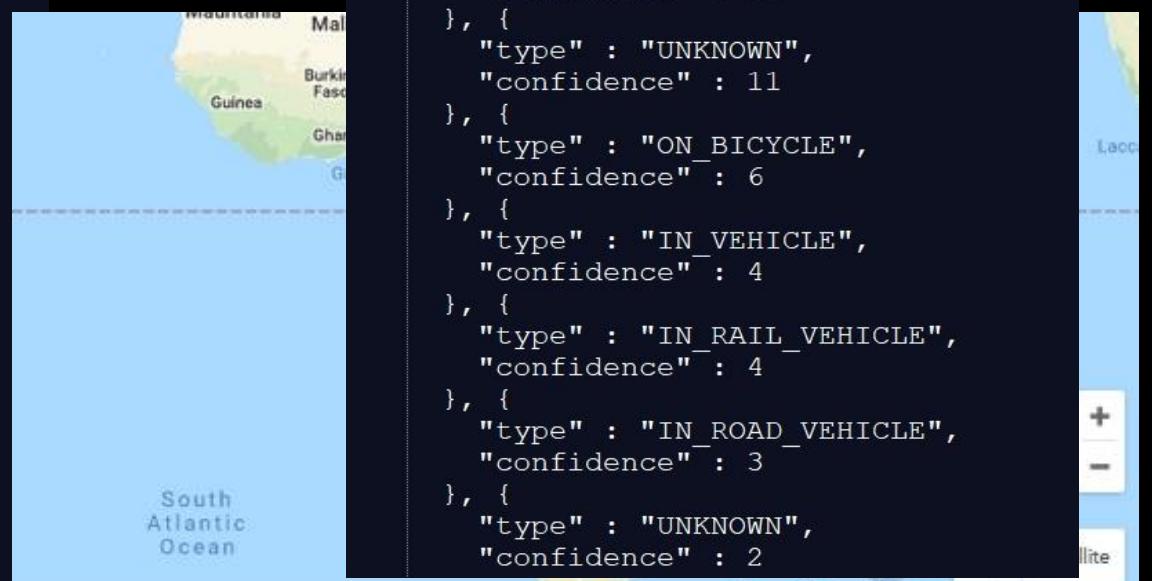
# google location history



Dailies, including specific activity types

# google location history

```
Location History.json
1  {
2    "locations" : [
3      {
4        "timestampMs" : "1534875901466",
5        "latitudeE7" : 136928408,
6        "longitudeE7" : -892377220,
7        "accuracy" : 82,
8        "altitude" : 810,
9        "verticalAccuracy" : 2,
10       "activity" : [
11         {
12           "timestampMs" : "1534875890588",
13           "activity" : [
14             {
15               "type" : "STILL",
16               "confidence" : 100
17             }
18           ]
19         },
20         {
21           "timestampMs" : "1534875875533",
22           "latitudeE7" : 136927999,
23           "longitudeE7" : -892377783,
24           "accuracy" : 32,
25           "altitude" : 810,
26           "verticalAccuracy" : 2
27         },
28         {
29           "timestampMs" : "1534875853593",
30           "latitudeE7" : 136927982,
31           "longitudeE7" : -892377634,
32           "accuracy" : 38,
33           "altitude" : 810,
34           "verticalAccuracy" : 2
35         }
36       ]
37     }
38   ]
39 }
```



Still working on these

# example - dhcp records

Search results (TownePlace) (1)					
		Type	Fields	Content	
^ Wireless Networks (1)					
<input checked="" type="checkbox"/>	1	Wireless Networks	SSID	 Wireless network: "TownePlace_GUEST"	

Galaxy S7 (SM-G930U)

Only one connected network, no BSSIDs or network records

# example - dhcp records

## Network Search

General Search    Network Detail

Set coordinates by address...

Address: saint george, ut



Last Observed: 20010925174546    Minimum data quality

SSID / Network Name (exact match): TownePlace\_GUEST

SSID / Network Name (wildcards<sup>1</sup>: % and \_): foobar%

Must Be a FreeNet     Must Be a Commercial Pay Net     Only

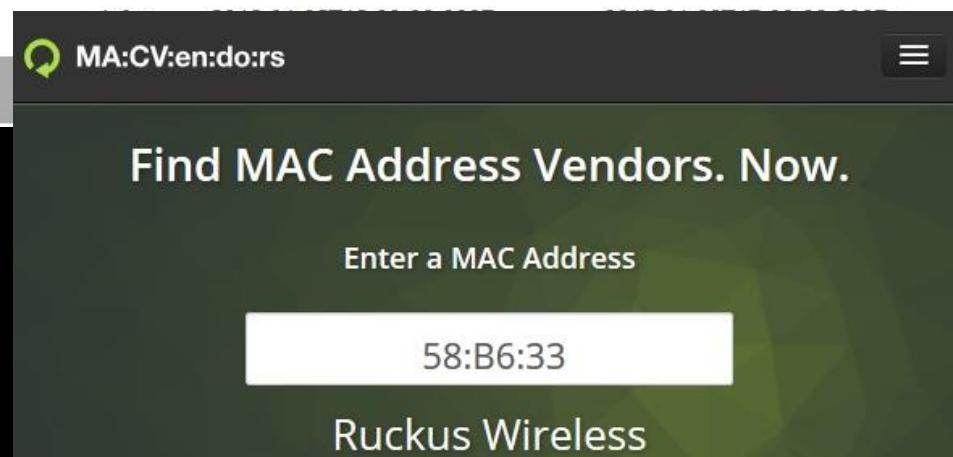
<sup>0</sup> 0-7 Product of number of observers and observations.  
<sup>1</sup> '%' means zero-or-more characters, '\_' means a single character.

Update Search Parameters

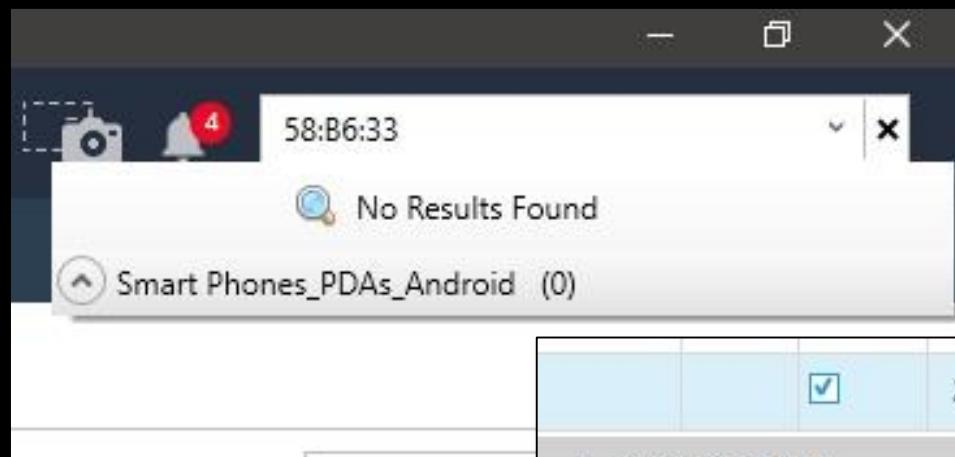
wigle search

# 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	-113.555113



# example - dhcp records



	<input checked="" type="checkbox"/>	23	<input checked="" type="checkbox"/>	Wireless Network...	2016-12-31 18:36:52(UTC-7)
^ 2017-01-25 (14)					
	<input checked="" type="checkbox"/>	24	<input checked="" type="checkbox"/>	Cell Tower: Locati...	2017-01-25 15:49:10(UTC-7)
	<input checked="" type="checkbox"/>	25	<input checked="" type="checkbox"/>	Cell Tower: Locati...	2017-01-25 15:49:50(UTC-7)
	<input checked="" type="checkbox"/>	26	<input checked="" type="checkbox"/>	Cell Tower: Locati...	2017-01-25 15:53:23(UTC-7)

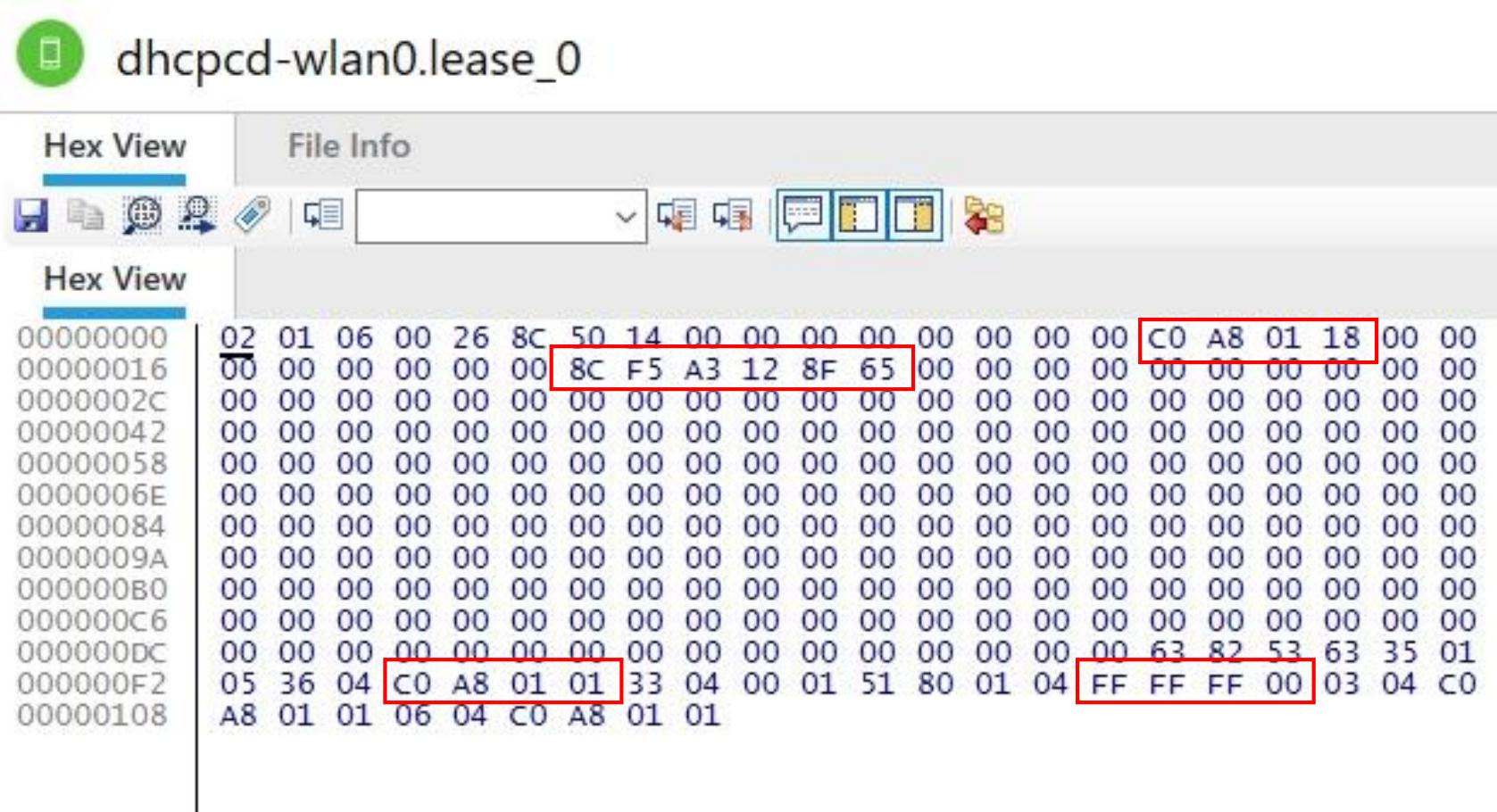
No results parsed for the particular networks

# 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

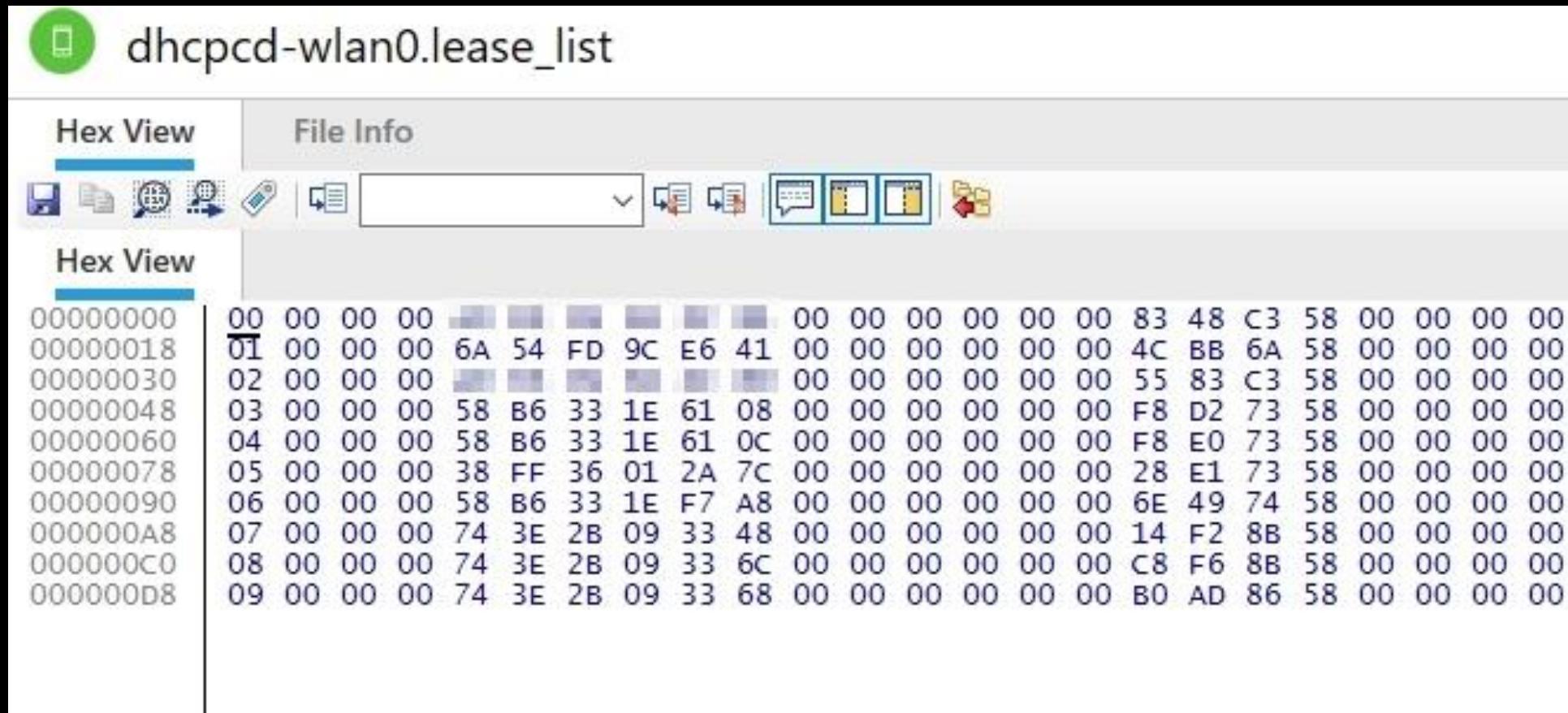


The screenshot shows a hex editor window titled "dhcpcd-wlan0.lease\_0". The "Hex View" tab is selected. The file contains a sequence of bytes, many of which are zero. Several specific byte sequences are highlighted with red boxes:

- A MAC address at offset 0x16: 8C F5 A3 12 8F 65
- A MAC address at offset 0x104: C0 A8 01 18
- A broadcast address at offset 0x104: FF FF FF 00
- A MAC address at offset 0x108: C0 A8 01 01

Offset	Value
00000000	02 01 06 00 26 8C 50 14 00 00 00 00 00 00 00 00
00000016	00 00 00 00 00 00 8C F5 A3 12 8F 65 00 00 00 00
0000002C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000042	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000058	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000006E	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000084	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000009A	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000C6	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000DC	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000F2	05 36 04 C0 A8 01 01 33 04 00 01 51 80 01 04 FF FF FF 00
00000108	A8 01 01 06 04 C0 A8 01 01 00 00 00 00 00 00 00 00

# example - dhcp records



The screenshot shows a hex editor interface with the title "dhcpd-wlan0.lease\_list". The "Hex View" tab is selected. The file contains 11 entries, each starting with a 4-digit index (00000000 to 000000D8) followed by a 16-byte hex dump. The data consists primarily of zeros, with some non-zero values appearing in specific fields like MAC addresses and IP addresses.

Index	Hex Data
00000000	00 00 00 00 6A 54 FD 9C E6 41 00 00 00 00 00 83 48 C3 58 00 00 00 00
00000018	01 00 00 00 6A 54 FD 9C E6 41 00 00 00 00 00 4C BB 6A 58 00 00 00 00
00000030	02 00 00 00 58 B6 33 1E 61 08 00 00 00 00 00 55 83 C3 58 00 00 00 00
00000048	03 00 00 00 58 B6 33 1E 61 08 00 00 00 00 00 F8 D2 73 58 00 00 00 00
00000060	04 00 00 00 58 B6 33 1E 61 0C 00 00 00 00 00 F8 E0 73 58 00 00 00 00
00000078	05 00 00 00 38 FF 36 01 2A 7C 00 00 00 00 00 28 E1 73 58 00 00 00 00
00000090	06 00 00 00 58 B6 33 1E F7 A8 00 00 00 00 00 6E 49 74 58 00 00 00 00
000000A8	07 00 00 00 74 3E 2B 09 33 48 00 00 00 00 00 14 F2 8B 58 00 00 00 00
000000C0	08 00 00 00 74 3E 2B 09 33 6C 00 00 00 00 00 00 C8 F6 8B 58 00 00 00 00
000000D8	09 00 00 00 74 3E 2B 09 33 68 00 00 00 00 00 B0 AD 86 58 00 00 00 00

dhcp\_lease\_list\_ext.py    mac\_mutate.py

## example - dhcp records

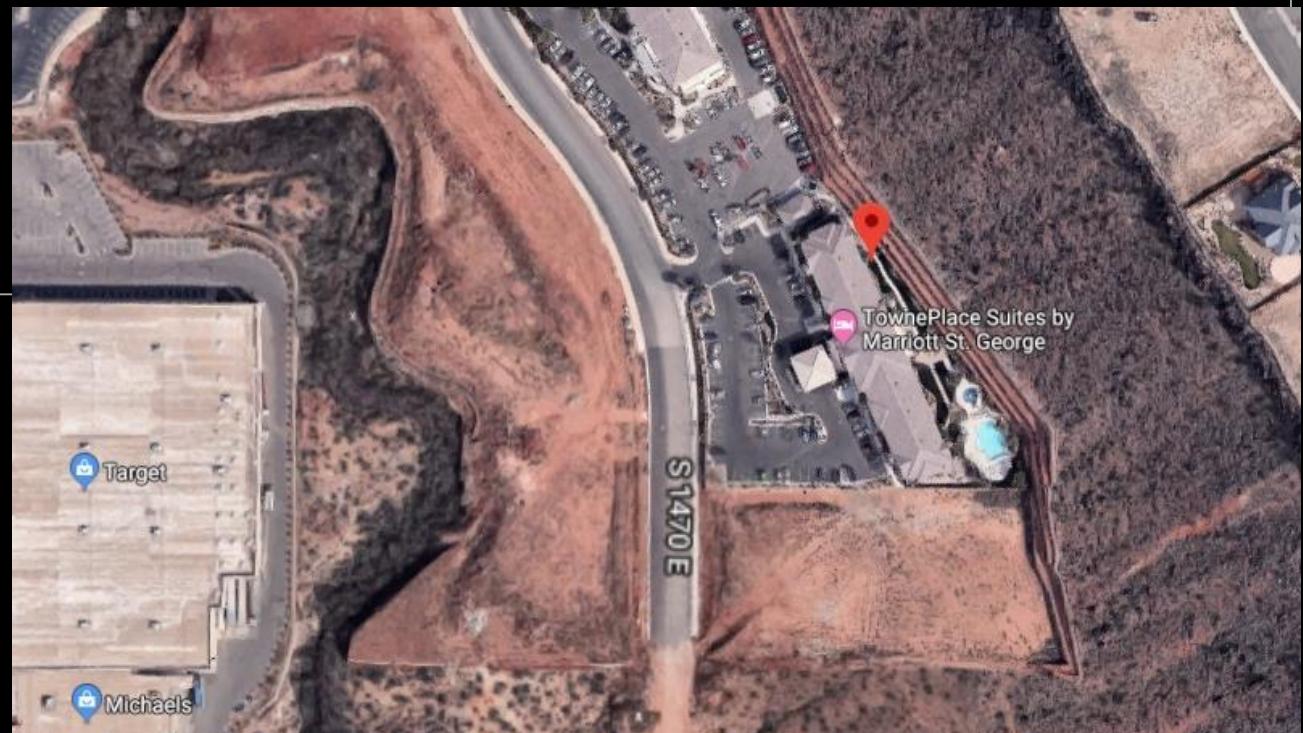
	A	B	C
1	DHCP Server MAC Address	TimeStamp	DHCP Server Manufacturer
2	[REDACTED]	2017-03-10 17:44:51	NETGEAR
3	6A:54:FD:9C:E6:41	2017-01-02 13:42:52	Amazon Technologies Inc.
4	[REDACTED]	2017-03-10 21:55:49	NETGEAR
5	58:B6:33:1E:61:08	2017-01-09 11:14:16	Ruckus Wireless
6	58:B6:33:1E:61:0C	2017-01-09 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

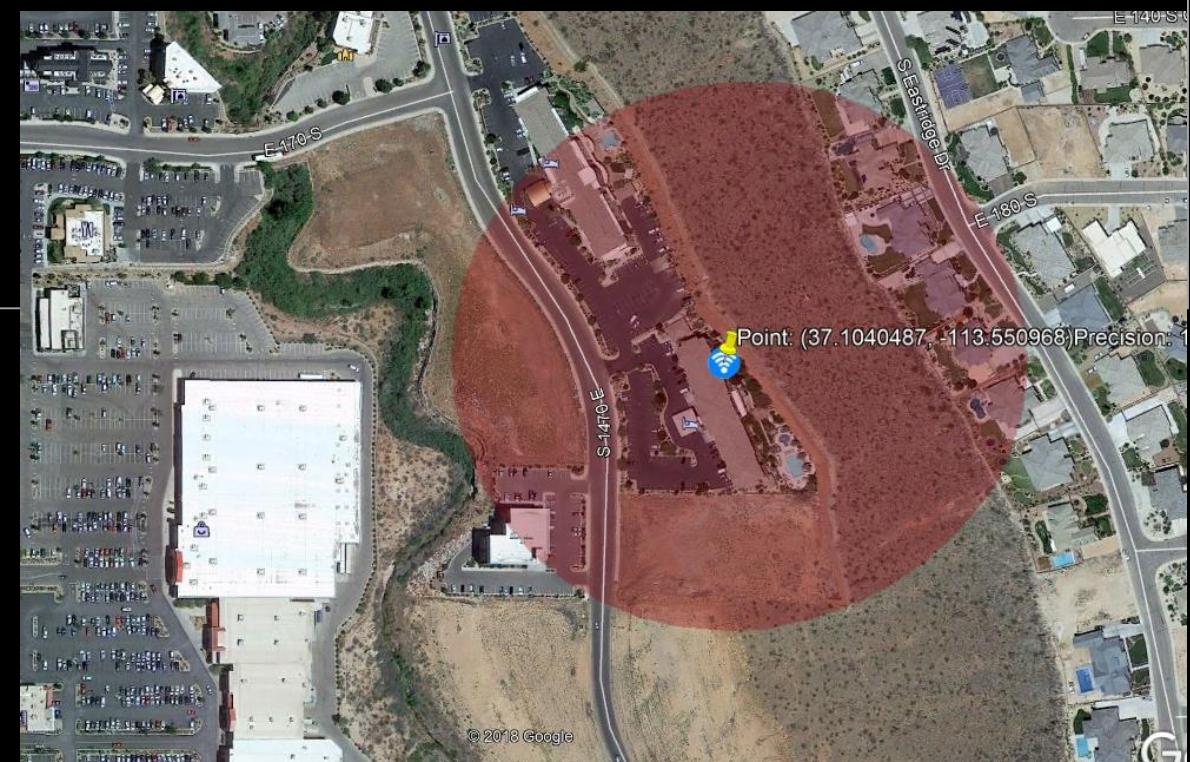
Latitude: 37.1040487
Longitude: -113.550968
Accuracy: 150.0 meters
Address: 251 S 1470 E, St. George, UT 84790, USA
```



# example - dhcp records

## Google Geolocation API search

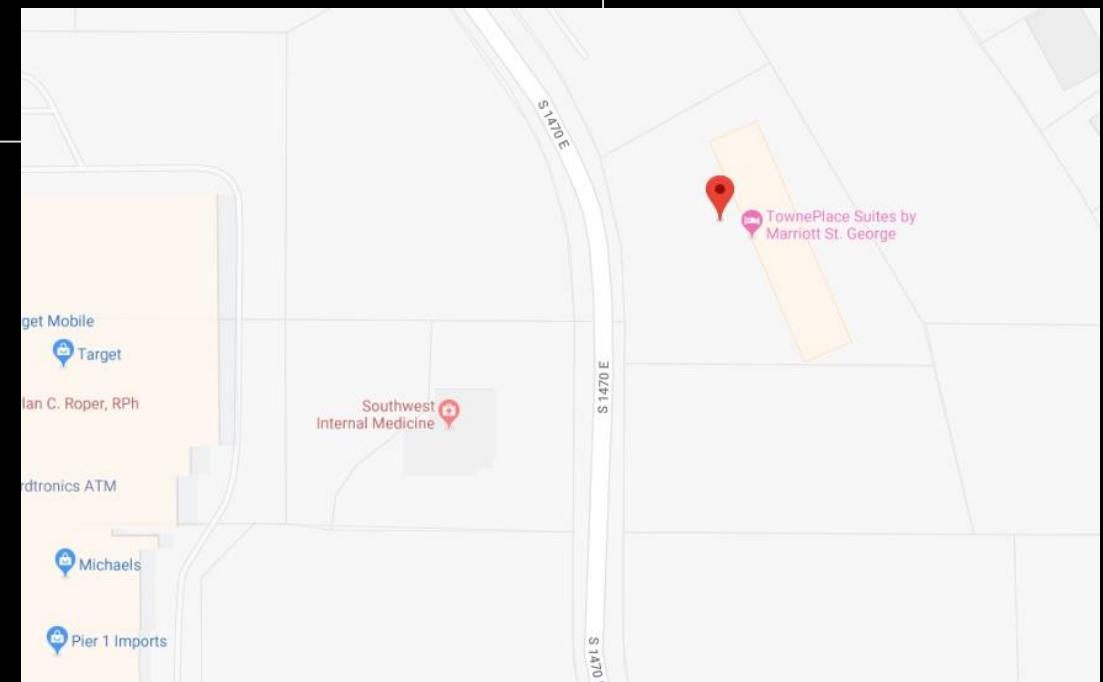
```
C:\Users\AARDV\Desktop\src\pfic_2018>python google2.py  
Enter macs (y/n): y  
Enter macs separated by comma only: 38:FF:36:01:2A:7C,58:B6:33:1E:61:08,58:B6:33:1E:61:0C,58:B6:33:1E:F7:A8  
MAC: 38:FF:36:01:2A:7C  
MAC: 58:B6:33:1E:61:08  
MAC: 58:B6:33:1E:61:0C  
MAC: 58:B6:33:1E:F7:A8  
  
Latitude: 37.1040487  
Longitude: -113.550968  
Accuracy: 150.0 meters  
Address: 251 S 1470 E, St. George, UT 84790, USA
```



# 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
```

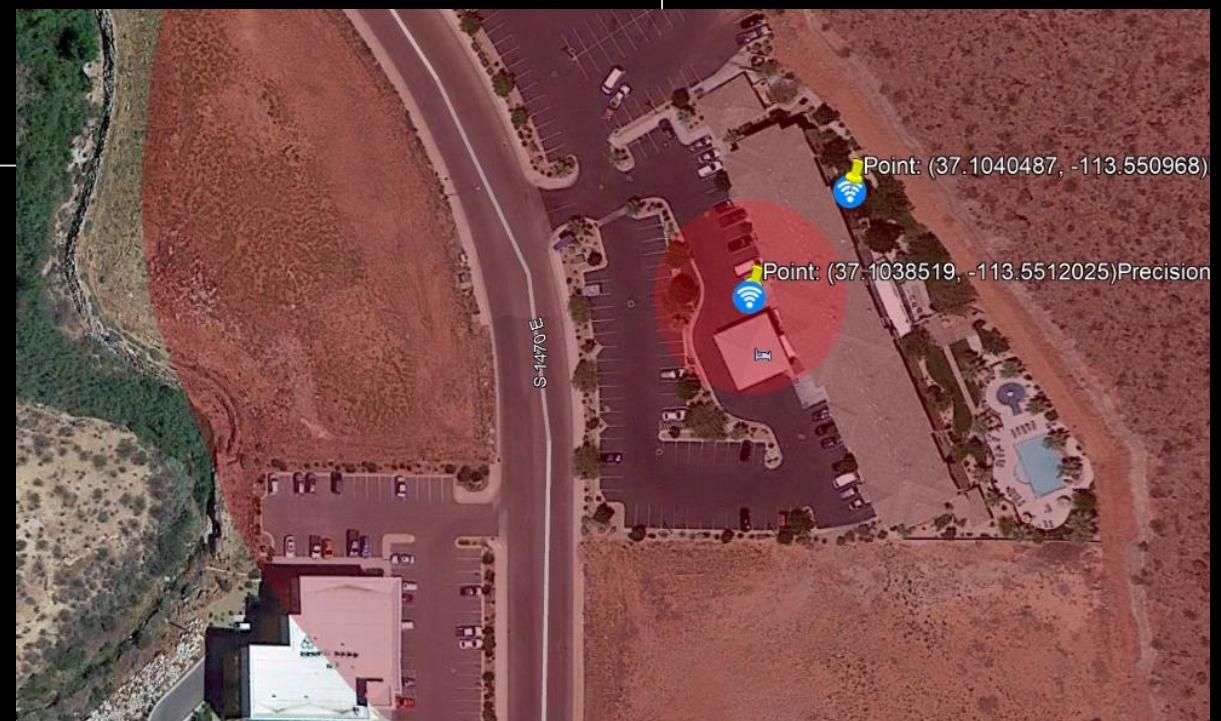


mozilla\_basic.py

# 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
```



# example - dhcp records

The screenshot shows a search results window titled "Search results (TownePlace) (1)". The results are categorized under "Wireless Networks (1)". There is one entry listed:

	#	Type	Fields	Content
<input checked="" type="checkbox"/>	1	Wireless Networks	SSID	Wireless network: "TownePlace_GUEST"

Below the search results, there are three collapsed dropdown menus, each showing "No Results Found". The dropdowns are labeled:

- Smart Phones\_PDAs\_Android (0)
- Smart Phones\_PDAs\_Android (0)
- Smart Phones\_PDAs\_Android (0)

At the bottom of the interface, there are three tabs labeled "Wireless Network" and "Go to".

# demo - lg dhcp records

```
dhpcd-wlan0.lease  
dhpcd-wlan0.pid  
dhpcd-wlan0-00_0c_e6_0b_83_01.lease2  
dhpcd-wlan0-00_0c_e6_dd_02_01.lease2  
dhpcd-wlan0-00_24_7b_71_0d_5c.lease2  
dhpcd-wlan0-2c_e6_cc_53_5f_88.lease2  
dhpcd-wlan0-2c_e6_cc_53_41_48.lease2  
dhpcd-wlan0-2c_e6_cc_d2_78_f8.lease2  
dhpcd-wlan0-2c_e6_cc_d2_80_48.lease2  
dhpcd-wlan0-6c_9c_ed_87_ea_1e.lease2  
dhpcd-wlan0-6c_9c_ed_87_ea_11.lease2  
dhpcd-wlan0-6c_aa_b3_11_d5_98.lease2  
dhpcd-wlan0-18_8b_9d_59_da_64.lease2  
dhpcd-wlan0-20_3a_07_9f_8b_64.lease2  
dhpcd-wlan0-20_3a_07_9f_85_fb.lease2  
dhpcd-wlan0-24_b6_57_f8_46_de.lease2  
dhpcd-wlan0-34_a8_4e_3a_3e_d4.lease2  
dhpcd-wlan0-40_f4_ec_7e_e8_01.lease2  
dhpcd-wlan0-64_a0_e7_88_2b_ce.lease2  
dhpcd-wlan0-64_a0_e7_90_13_51.lease2
```

```
dhpcd-wlan0-84_18_3a_0f_9b_48.lease2  
dhpcd-wlan0-84_18_3a_0f_9c_a8.lease2  
dhpcd-wlan0-84_18_3a_0f_9c_ac.lease2  
dhpcd-wlan0-84_18_3a_0f_b5_bc.lease2
```

A	B	C	D	E	F	G	H	I	J
ssid	bssid	lat	lng	lasttime	country	region	city	house_number	road
1	gogoinflight	00:0C:E6:0B:83:01	42.06707001	-94.48470306	2018-08-18T19:00:00.000Z	US	IA		202nd Street
2	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

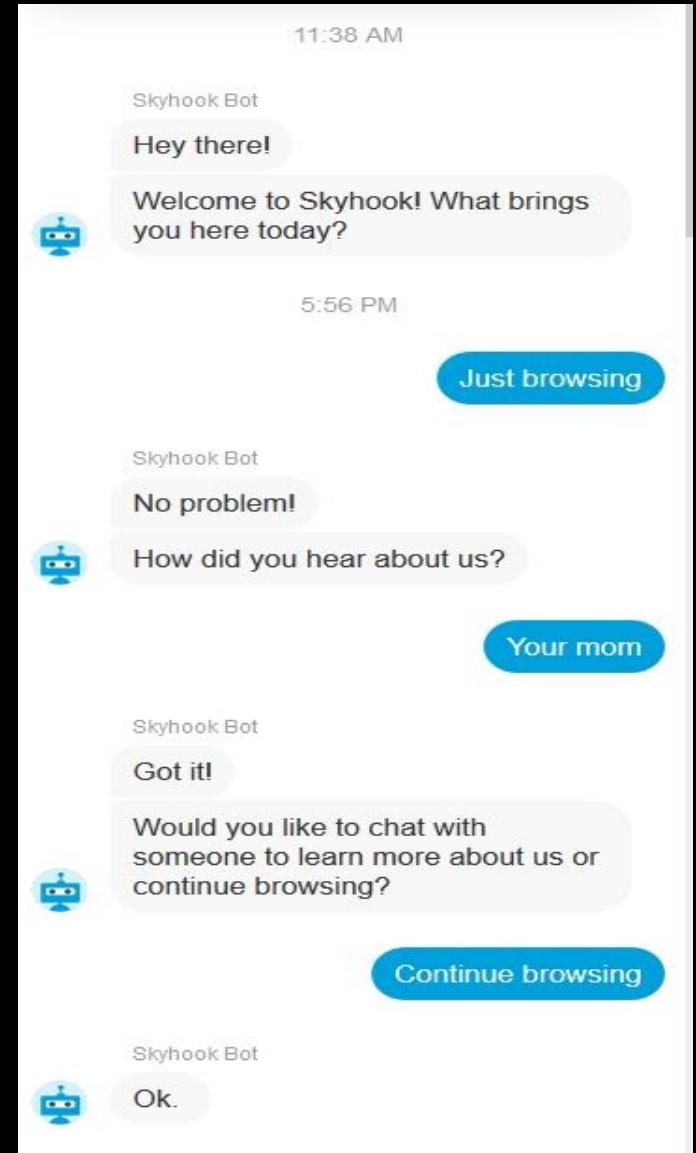
- [wigle.net](http://wigle.net)
- [opencellid.org](http://opencellid.org)
- <https://location.services.mozilla.com/>
- <https://www.openstreetmap.org>
- <https://www.gpsbabel.org/>
- <https://www.darrinward.com/lat-long/>
- <http://www.earthpoint.us/Convert.aspx>
- <https://github.com/googlemaps/google-maps-services-python>

# 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, wardriving
- 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](mailto:kim@h11dfs.com)
- [https://github.com/gkt-aardvark/geo\\_location\\_tools](https://github.com/gkt-aardvark/geo_location_tools)
- @ArdJect on Twitter