Detecting Bluetooth Surveillance Systems

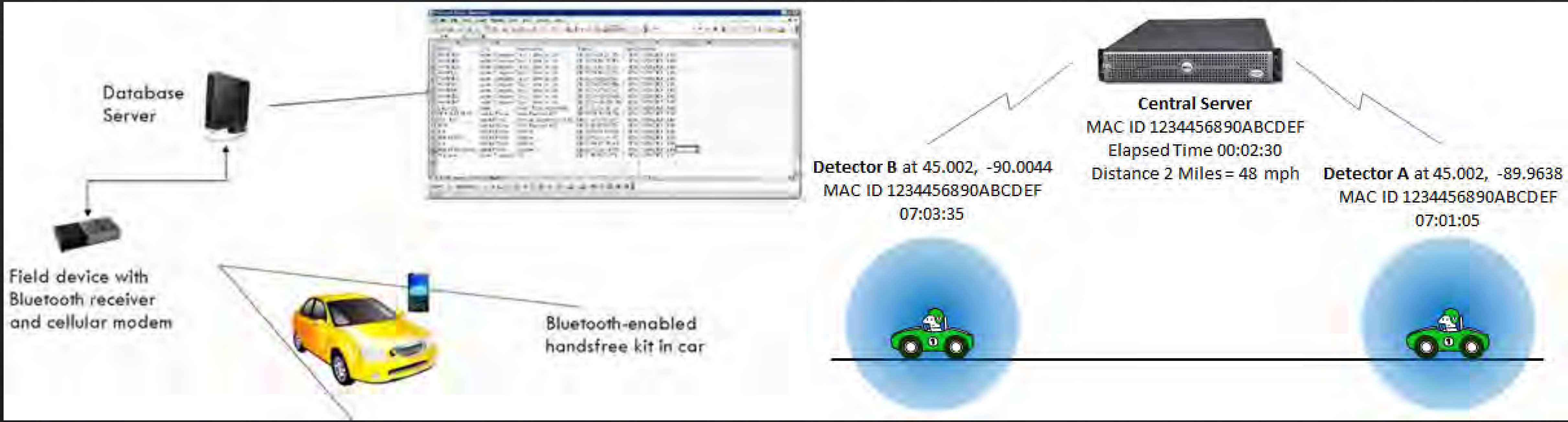
Grant Bugher

http://perimetergrid.com

DEFCON 101 Track @ DEFCON 22

They’re Already Watching

Federal Highway Administration



John Shaw, Wisconsin DOT, 2013

Installing Bluetooth Detectors

Form Factors

Federal Highway Administration, 2013

Seattle Experiments

Research Note

Travel Time Data Obtained from Bluetooth Technology

From the WSDOT Research Office
December 2011

Disclaimer: The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the Washington State Department of Transportation or Federal Highway Administration. This report does not constitute a standard, specification, or regulation.

WSDOT Travel Information

Introduction

Roadway users easily understand travel time, and consider it one of the most important transportation metrics. Users often get travel time directly through dynamic message signs (DMS), 511 phone services, and online systems that

and the Internet, allowed researchers to develop a MAC address-based tracking method. This method relies on recording the MAC addresses of bypassing devices at one location and noting the time difference

cost of the Bluetooth reader, and that the MAC address collection device spans multiple lanes. This provides a significant advantage compared to Automatic License Plate Recognition (ALPR) systems that require lane-based detection to determine travel times. Additionally, Bluetooth-based travel time data collection systems are easy to install and do not require high bandwidth for communications. Compared to Global Positioning Systems (GPS), the MAC address-based systems do not require willing volunteers with vehicles equipped to provide constantly recorded GPS coordinates. Instead, all surrounding devices get free broadcast of the MAC address. Users not wishing to disclose their location can simply turn off the broadcast function on their MAC devices.

Although the MAC address-based collection techniques have significant advantages, there are some drawbacks to their use. Relatively small sample

THE NEWS TRIBUNE

HOME | NEWS | SPORTS | BUSINESS | OPINION | LIVING & ENTERTAINMENT | LOCAL DEALS

News > Local News > **Military News**

State to monitor JBLM traffic congestion with Bluetooth

BY BRYNN GRIMLEY
Staff writer January 31, 2014

If you drive Interstate 5 from Lacey to Tacoma and have a Bluetooth device, chances are you're going to be tracked in the next couple of weeks.

No, it's not the National Security Agency monitoring your every move. It's the state Department of Transportation.

The tracking of Bluetooth devices — or, more accurately, the unique address they emit — is part of the state's effort to study congestion on the I-5 corridor around Joint Base Lewis-McChord.

For those unnerved by the idea of being tracked, transportation officials say there's nothing to worry about. "We cannot track people or hear conversations, nor do we want to. All we know is the signal," said Jon Pascal, principal at Transpo Group, the Kirkland-based company deploying the

Why Bluetooth?

Bluetooth vs. Loops, Radar, ALPR, and Pilot Cars

- More data
  - Loops gather volume only
  - Radar gathers volume and speed
  - Bluetooth provides volume, speed, and route choice
- Inexpensive
  - $2000-5000 per detector
  - Cellular modem & solar power with battery back-up or PoE
- Limitations
  - 3-6% sample rate, plus trucks oversampled
  - Sensitive to site characteristics & antenna placement
- Fewer privacy concerns?

Information Leakage

- Frequency-hopping pattern derived from MAC address

Non-Significant Address Part (NAP)	Upper Address Part (UAP)	Lower Address Part (LAP)
00:00	09	4E:22:19
Manufacturer-assigned, irrelevant	Can be derived from traffic	Sent in every packet

- Any device can send an inquiry from the GIAC anonymous address, 9E:8B:33:16
  - Devices set Discoverable will respond from their MAC, revealing LAP
- Any device that knows your MAC can initiate a connection
  - Connection attempts always get device name, class, service list, and clock offset
  - Can also sniff traffic with UAP+LAP, but it’s encrypted

Scanning Bluetooth

- Inquiry-based tracking
  - Scanner just spams Bluetooth inquiries
  - Gets all *discoverable* stations in range every 10.48 seconds
- Connection-based tracking
  - Scanner keeps paging for ACL and L2CAP connections
  - Works without discoverability, but only for known LAPs
- Passive sniffing
  - Scanner sweeps the Bluetooth frequency range and gathers sampled packets
  - Works without discoverability, provides only LAP, can’t sniff data
  - Requires nonstandard hardware, not OTS Bluetooth transceivers

Seattle Deployment

Figure 4-1: a) Selected freeway test corridor on SR 520. b) Bluetooth sensor (left) and portable ALPR (right) used to collect travel time data at the 24th Ave location.

will be in use. Such devices could include a valve-cap pressure sensor that broadcasts tire pressure to the main in-vehicle computer system or a temperature sensor on the vehicle windshield. Knowing the types of devices present may allow for finer-grained analysis, such as distinguishing between passenger vehicle and truck travel times.

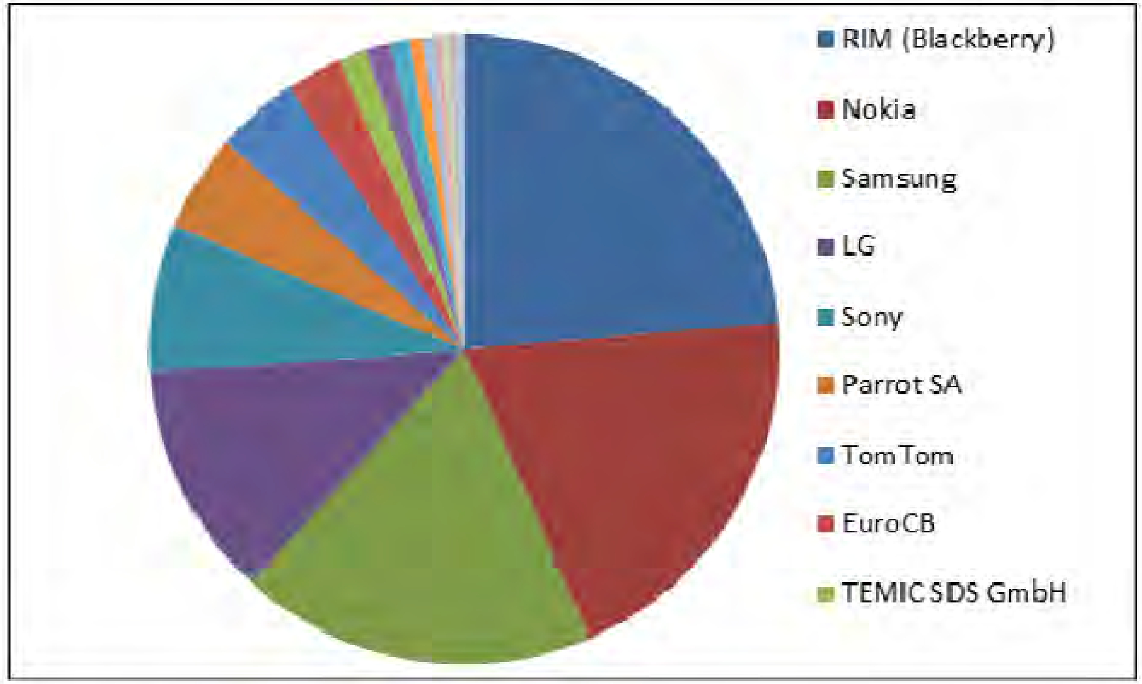


Figure 4-10: Relative shares of detected device manufacturers

Privacy Concerns?

Privacy protection and filtration

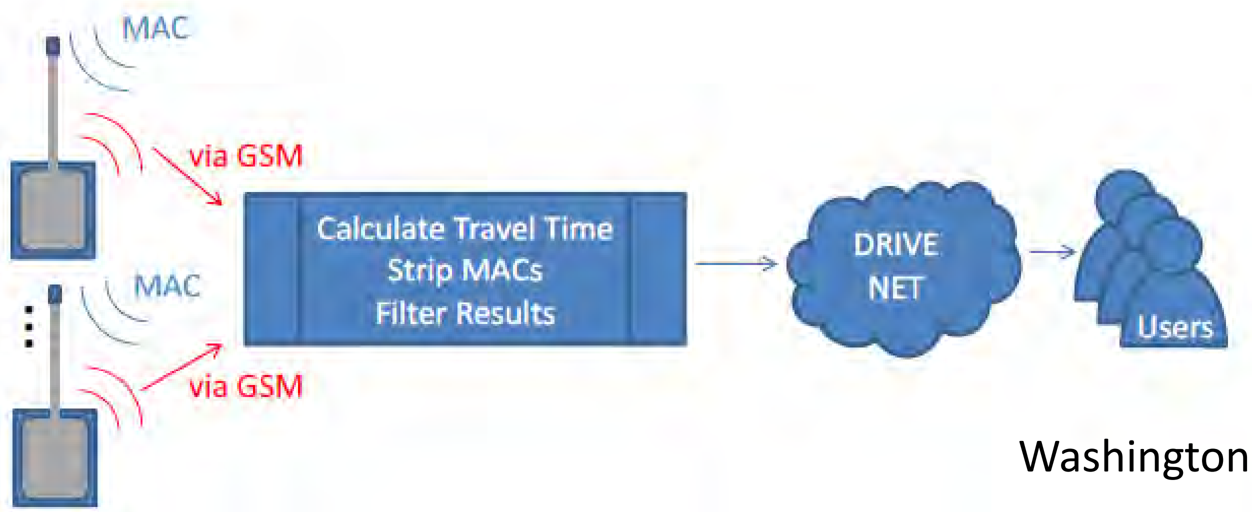
Of course, Bluetooth devices within vehicles might not be the only ones to pass a reader – pedestrians, transit riders, etc, could all be in possession of their own Bluetooth-enabled technologies. Will these not influence readings? "Our host software uses various statistically based algorithms to filter matches that appear to be outliers," explains TTI's Tony Voigt. "These algorithms can be configured based on the characteristics of each individual segment being monitored."

The research engineer admits the distinction of transit vehicles versus other vehicles and pedestrians is difficult in practice, but possible using TTI's in-house-developed field software processes. "Much of our intellectual property is based on this. We have seen gains in matches of 50% over other processes with our patent-pending methods, which allow for a more robust analysis capability, including potentially differentiating transit vehicles.

A further critical aspect of the TTI process is privacy protection. "We have the capability to make the MAC address data collected anonymous before transmitting from the field, which we can do without any reduction in the fidelity of the data." But isn't anonymity a major benefit of Bluetooth? If MAC addresses are not linked to a user, why the extra process? "If there is even a very small chance that a hacker could sniff the communications pathway from field to host, there should be procedures and protocols in place to minimize the threat."

Voigt says anonymizing the data may be more of a benefit if the raw data is archived for later analysis. A partial MAC address when anonymized then archived is less subject to scrutiny, although there are methods to use to enable further use of the data for operational and planning purposes, such as higher-level origin/destination studies.

TTI's AWAM can distinguish 'groups' of Bluetooth devices on a particular route functioning properly and has not been tampered with. Once the synchronization and location recording are complete, the device begins data collection, recording the bypassing MAC addresses and their respective timestamps. As data are collected, they are sent over the GSM network to a server in STAR Lab, where the MACs are kept for a specified period (currently 60 minutes). If a matching MAC is received during this period, a travel time is calculated, the MAC address is deleted, and the data are uploaded to the Digital Roadway Interactive Visualization and Evaluation Network (DRIVE Net) developed by the STAR Lab for data sharing, modeling, and online analysis (Ma et al., 2011). This approach to data collection allows real-time information to flow to users while maintaining a level of privacy. Figure 3-3 illustrates the overarching structure of the data collection effort.



Washington State Transportation Center, 2011

No Privacy Concerns

Canadian Border Deployment

Figure 4-5: SR 539 and Badger Rd. Bluetooth installation location

Figure 4-6: SR 539 U.S.-Canada border Bluetooth installation location

University of Washington STAR Lab

Privacy Mitigations

- Capturing LAP only
  - No central registry of Bluetooth MACs to devices or purchasers
  - LAP is 24 bits, not globally unique
- Anonymizing LAP
  - Needs to remain unique
  - Hash before storing; salted hash is possible
- Limited Retention
  - UW STAR Lab experiment limited to 60 minutes
  - Commercial readers often limit to 20 minutes

LAP only?

will be in use. Such devices could include a valve-cap pressure sensor that broadcasts tire pressure to the main in-vehicle computer system or a temperature sensor on the vehicle windshield. Knowing the types of devices present may allow for finer-grained analysis, such as distinguishing between passenger vehicle and truck travel times.

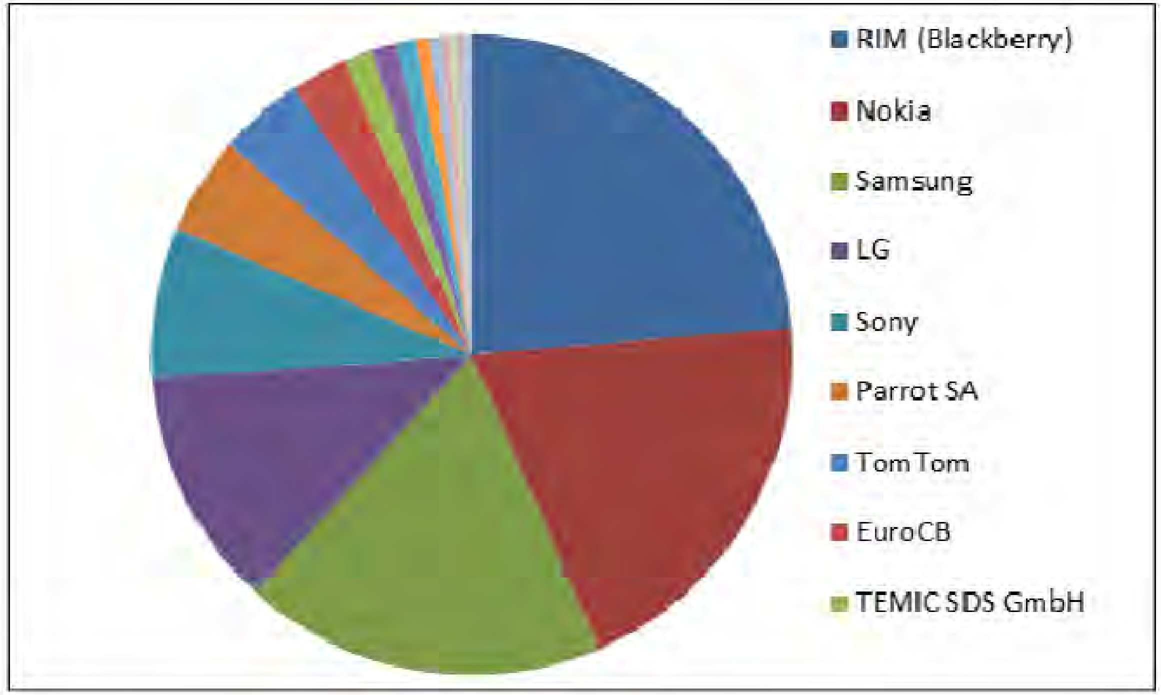


Figure 4-10: Relative shares of detected device manufacturers

- Remember this slide?

  LAP does not permit manufacturer identification; UAP is captured

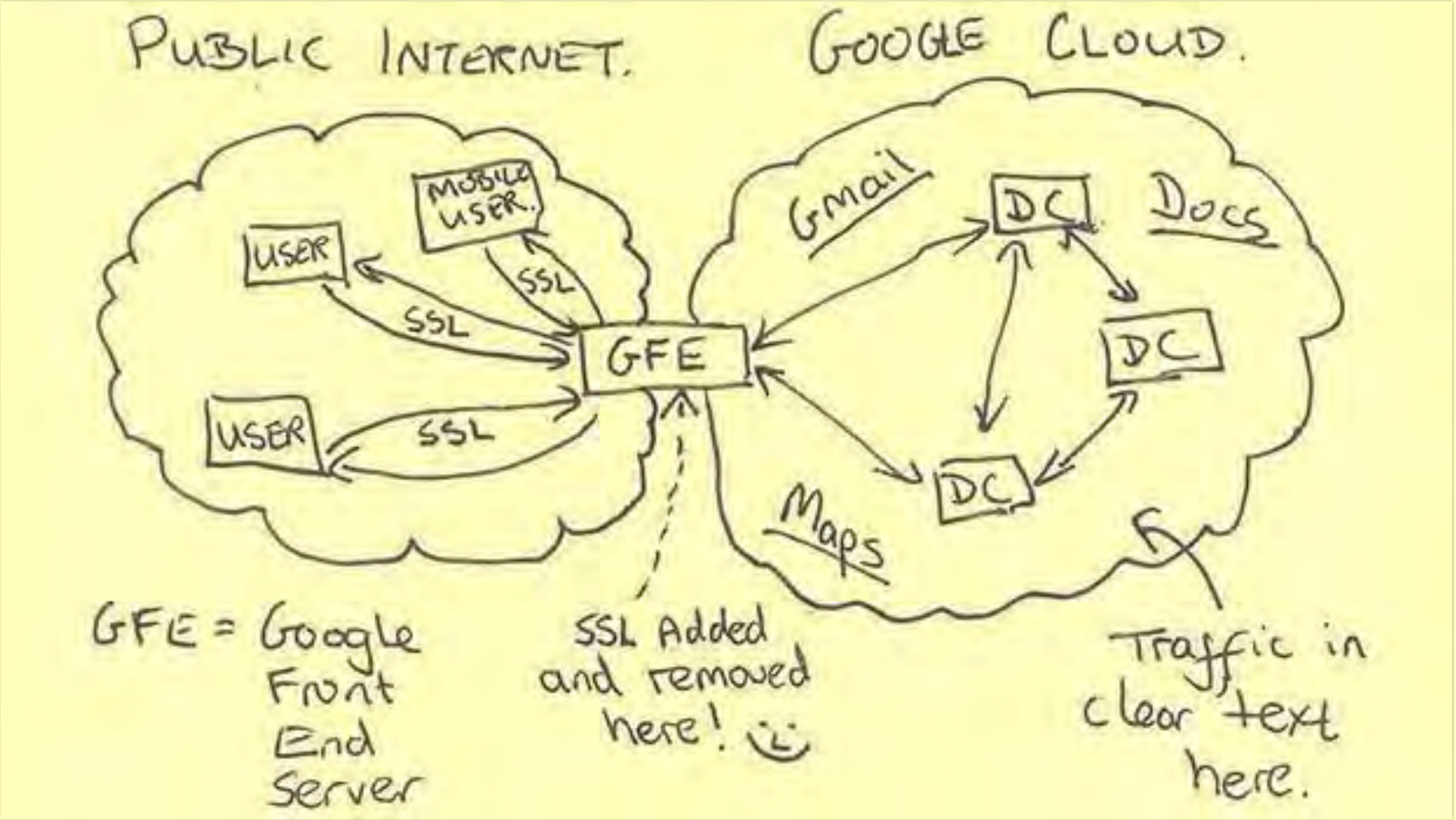
- Not globally unique?

  1 in 24,048,576 is pretty unique in most metro areas.

  Have two or more devices in your car?

Anonymizing LAP?

- There are 8×10^{67} ways to shuffle a deck of cards, which is about 225 bits of entropy
- If you do it by calling rand(), there's only 4×10^{9} ways to shuffle a deck of cards, 32 bits of entropy.
- Taking 256-bit salted hashes of a 24-bit LAP is useless.

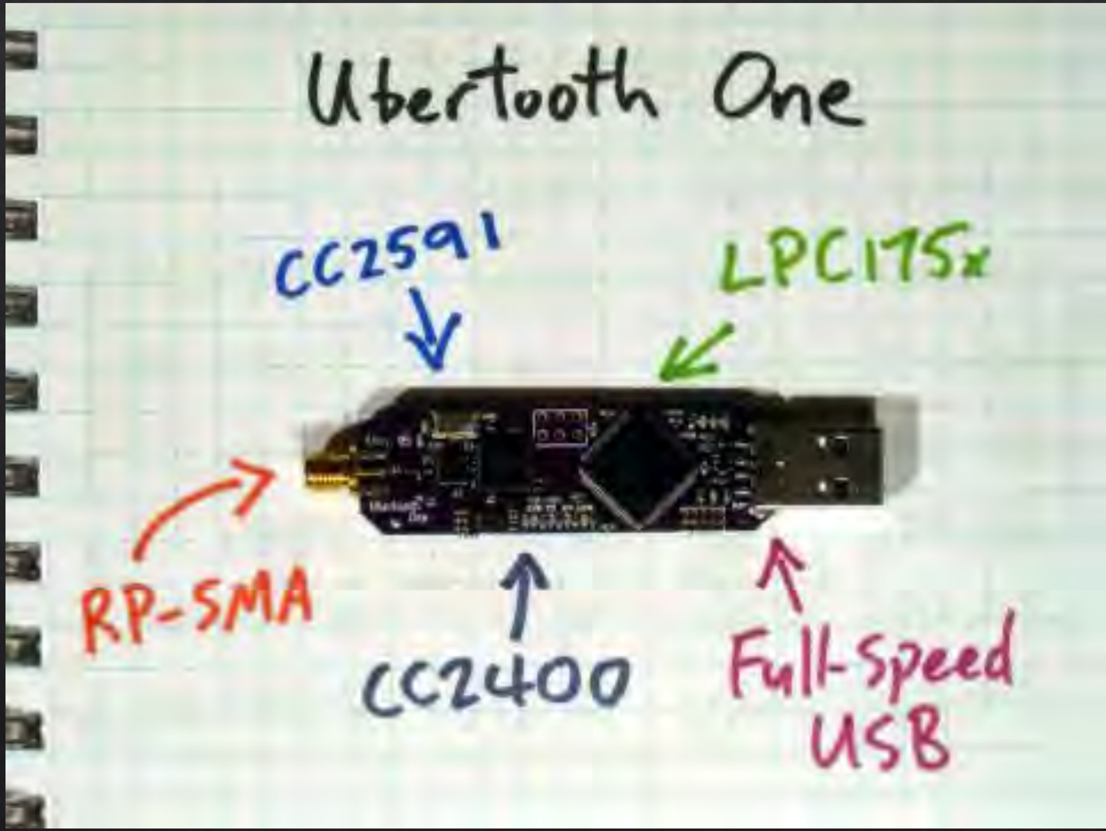


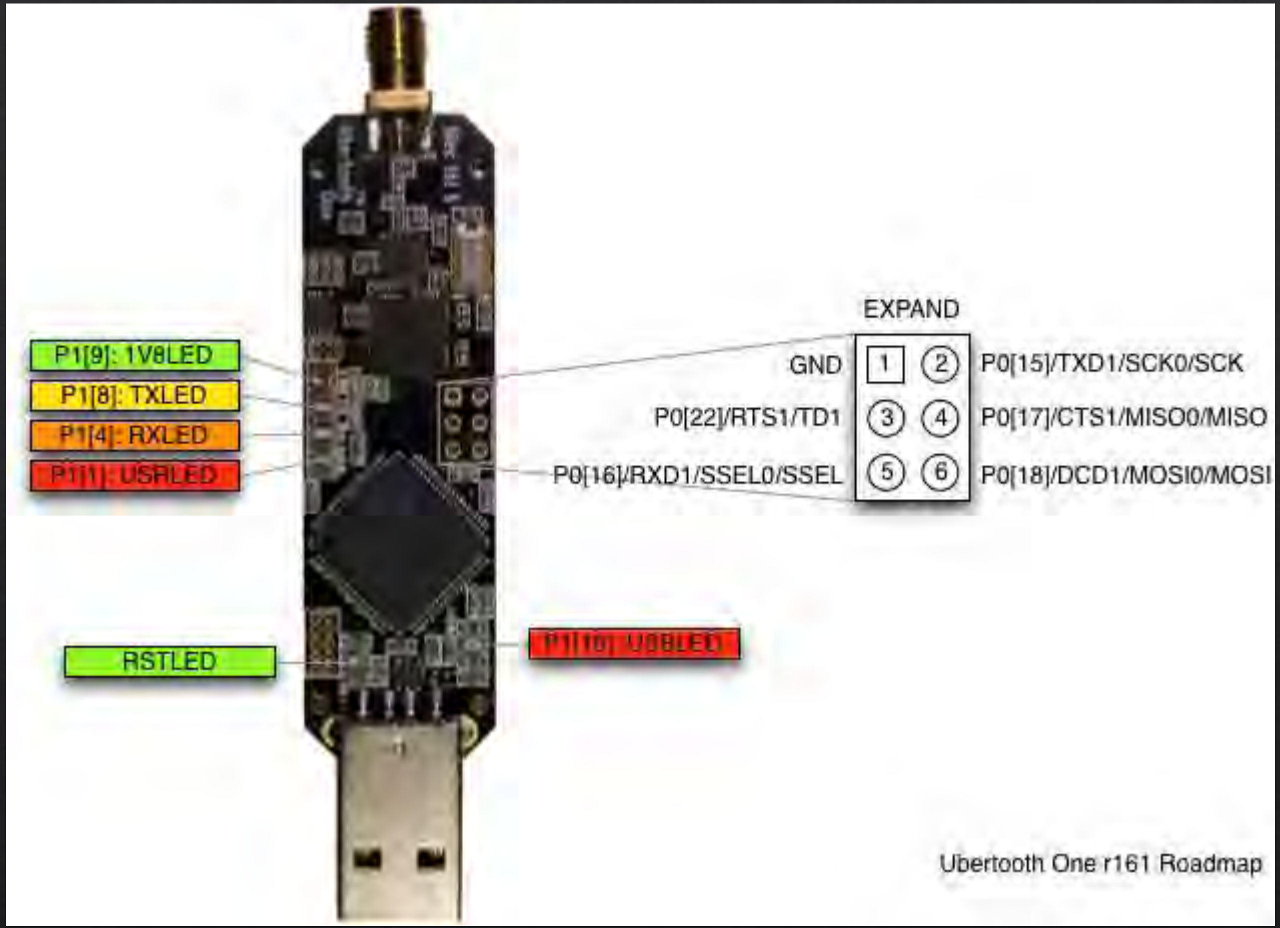
Edward Snowden by way of the Washington Post

Making our Own

Bluetooth Scanning for Fun and Profit

- Bluetooth transceivers can send and receive, but only beacons or when paired
  - Class 1: 100 mW, 100m range
  - Class 2: 2.5 mW, 10m range
  - Class 3: 1 mW, 1m range
- Commercial Bluetooth "sniffers" can grab all channels for the low price of $10,000
  - Receive-only, no transmission
  - Still single- or dual-antenna "sweep," not true all-channel receive


Ubertooth One
CC2591
LPC175x
RP-SMA
CC2400
Full-Speed USB


EXPAND
P1[9]: 1V8LED
P1[8]: TXLED
P1[4]: RXLED
P1[1]: USRLED
GND
P0[15]/TXD1/SCK0/SCK
P0[22]/RTS1/TD1
P0[17]/CTS1/MISO0/MISO
P0[16]/RXD1/SSEL0/SSEL
P0[18]/DCD1/MOSI0/MOSI
RSTLED
Ubertooth One r161 Roadmap

PC-Based Bluetooth Scanner

- Linux required, no Windows Ubertooth drivers exist
  - Can use a Linux VM
  - Need WinUSB driver associated with Ubertooth to pass it through to the VM
- Equipment:
  - Old ThinkPad in the garage, or a VirtualBox VM, $0
  - Ubuntu 12.04.4 LTS 64-bit Desktop, $0
  - Ubertooth One, $119 or build your own
  - GlobalSat ND-100S USB GPS Dongle, $37
  - Azio BTD-V201 USB Micro Bluetooth Adapter, Class 1, V2.1 + EDR, $16

```
$ sudo apt-get update; sudo apt-get upgrade

$ sudo apt-get install libusb-1.0-0-dev libbluetooth-dev libusb-dev
libpcap-dev libpcap0.8-dev libcap-dev pkg-config libnl-dev libncurses5-
dev libpcre3-dev gpsd gpsd-clients python-gps bluez python-pyside.qtgui

$ wget https://github.com/walac/pyusb/archive/1.0.0b1.tar.gz

$ wget https://github.com/greatscottgadgets/libbtbb/archive/2014-02-
R2.tar.gz

$ wget https://github.com/greatscottgadgets/ubertooth/archive/2014-02-
R2.tar.gz

$ wget https://www.kismetwireless.net/code/kismet-2013-03-R1b.tar.xz
```

Make and install all of that. Use the non-phyneutral plugin. See Ubertooth Wiki if you need help.

Reboot or `$ sudo ldconfig` before running tools.

ARM-Based Bluetooth Scanner

- Raspberry Pi is an inexpensive ($35) open-source hardware platform
  - Based on ARM architecture, Runs on 5V USB power
  - 2 USB ports, HDMI, audio, GPIO pins with SPI capability
- Equipment:
  - Raspberry Pi, $40
  - Adafruit PiTFT minkit (optional), $35
  - Adafruit Pibow enclosure (optional), $22
  - Gear from PC build: Ubertooth One, GlobalSat ND-100S USB GPS Dongle, Azio BTD-V201 USB Micro Bluetooth Adapter, Class 1, V2.1 + EDR
  - 5V 500 mA USB power source: car charger or Gorilla Gadgets External Battery Pack, $29
  - Powered USB hub, with power

Install latest Raspbian image. If using PiTFT, use Adafruit image instead. Configure locales, enable SSH.

```
$ sudo rpi-update; sudo apt-get update; sudo apt-get upgrade
```

```
$ sudo apt-get install libusb-1.0-0-dev libbluetooth-dev libusb-dev libpcap-dev libpcap0.8-dev libcap-dev pkg-config libnl-dev libncurses5-dev libpcre3-dev gpsd gpsd-clients python-gps bluez python-pyside.qtgui
```

```
$ wget https://github.com/walac/pyusb/archive/1.0.0b1.tar.gz
```

```
$ wget https://github.com/greatscottgadgets/libbtbb/archive/2014-02-R2.tar.gz
```

```
$ wget https://github.com/greatscottgadgets/ubertooth/archive/2014-02-R2.tar.gz
```

```
$ wget https://www.kismetwireless.net/code/kismet-2013-03-R1b.tar.xz
```

Make and install all of that. Use the non-phyneutral plugin. See Ubertooth Wiki if you need help.

Reboot or `sudo ldconfig`

```
$ sudo usermod -G kismet,plugdev -a pi
```

In ~/ubertooth/host/libubertooth, modify 40-ubertooth.rules to change "usb" to "plugdev"

```
$ sudo cp ~/ubertooth/host/libubertooth/40-ubertooth.rules
/etc/udev/rules.d

$ sudo adduser scanner

$ sudo usermod scanner -a -G
adm,dialout,video,plugdev,users,netdev,input,spi,gpio,kisme
```

Modify /etc/inittab:

```
-"1:2345:respawn:/sbin/getty --noclear 38400 tty1"
+"1:2345:respawn:/bin/login -f scanner tty1 </dev/tty1 >/dev/tty1 2>&1"
```

```
$ sudo dpkg-reconfigure -plow gpsd
```

Modify /etc/default/gpsd:

```
GPSD_OPTIONS="/dev/ttyUSB0"
DEVICES=""
START_DAEMON="true"
SBAUTO="true"
GPSD_SOCKET="/var/run/gpsd.sock"
```

Run `$ kismet -c ubertooth`, enable ubertooth_ui plugin and configure UI

Modify .bashrc:

```
+"kismet -c ubertooth"
```

Issues

- Raspberry Pi power issues
  - Pi USB ports provide only 140mA, not enough for Ubertooth & GPS
  - Powered USB hub, modified to get power from USB, solves this problem
- Ubertooth + Kismet issues

  *** glibc detected *** kismet_server: corrupted double-linked list: 0x00ce8348 ***

  kernel: [4474.980885] kismet_server[6253]: segfault at 131ffb618 ip 00007f2e31cedbe6 sp 00007fffbf5a25a0 error 4 in libc-2.13.so[7f2e31c74000+182000]

  - Works flawlessly on Ubuntu 12.04.4 LTS and Ubuntu 14.04
  - Issue occurs on Raspbian (all versions) and Kali 1.0.7
  - Known issue under investigation; thanks, Dominic Spill!

Scanning

- ubertooth-scan
  - Performs an inquiry scan, as we suspect the DoT does
  - Requires both Ubertooth One and a functioning Bluetooth transciever
  - Takes 10.24 seconds to be sure of catching all devices
  - Reports LAP only
- kismet -c ubertooth
  - Performs passive monitoring on all frequencies
  - Requires Ubertooth One only
  - Continuous sweep, will miss things
  - Reports LAP always, UAP with enough data
  - Bluetooth and GPS logging

Results

1	No.	Time	LAP	Lat	Long
91	93105	7608.17	00:00:9e_8b:33:16	47.62956	-122.159
92	93106	7608.183	00:00:9e_8b:33:16	47.62956	-122.159
93	101423	9435.235	00:00:9e_8b:33:16	47.76498	-122.076
94	101424	9435.63	00:00:9e_8b:33:16	47.76498	-122.076
95	101428	9442.926	00:00:9e_8b:33:16	47.76498	-122.076
96	101432	9444.304	00:00:9e_8b:33:16	47.76498	-122.076
97	101433	9445.298	00:00:9e_8b:33:16	47.76498	-122.076
98	101434	9445.507	00:00:9e_8b:33:16	47.76498	-122.076
99	101435	9445.712	00:00:9e_8b:33:16	47.76498	-122.076
100	101436	9445.732	00:00:9e_8b:33:16	47.76498	-122.076
101	112180	12489.53	00:00:9e_8b:33:16	47.76498	-122.076
102	112189	12490.1	00:00:9e_8b:33:16	47.76498	-122.076
103	112190	12490.12	00:00:9e_8b:33:16	47.76498	-122.076
104	112233	12491.9	00:00:9e_8b:33:16	47.76498	-122.076
105	112237	12492.09	00:00:9e_8b:33:16	47.76498	-122.076
106	112238	12492.11	00:00:9e_8b:33:16	47.76498	-122.076
107	112240	12492.13	00:00:9e_8b:33:16	47.76498	-122.076
108	112241	12492.16	00:00:9e_8b:33:16	47.76498	-122.076
109	112243	12492.37	00:00:9e_8b:33:16	47.76498	-122.076
110	112247	12492.5	00:00:9e_8b:33:16	47.76498	-122.076
111	112248	12492.51	00:00:9e_8b:33:16	47.76498	-122.076
112	112250	12492.69	00:00:9e_8b:33:16	47.76498	-122.076
113	112251	12492.71	00:00:9e_8b:33:16	47.76498	-122.076
114	112252	12492.72	00:00:9e_8b:33:16	47.76498	-122.076
115	121637	12878.07	00:00:9e_8b:33:16	47.76498	-122.076
116	121645	12878.44	00:00:9e_8b:33:16	47.76498	-122.076
117	121646	12878.44	00:00:9e_8b:33:16	47.76498	-122.076
118	121680	12880.18	00:00:9e_8b:33:16	47.76498	-122.076
119	121681	12880.2	00:00:9e_8b:33:16	47.76498	-122.076
120	162764	14996.63	00:00:9e_8b:33:16	47.76498	-122.076
121	162769	14996.97	00:00:9e_8b:33:16	47.76498	-122.076
122	162770	14996.98	00:00:9e_8b:33:16	47.76498	-122.076
123	162785	14997.41	00:00:9e_8b:33:16	47.76498	-122.076
124	162786	14997.43	00:00:9e_8b:33:16	47.76498	-122.076
125	162801	14998.18	00:00:9e_8b:33:16	47.76498	-122.076

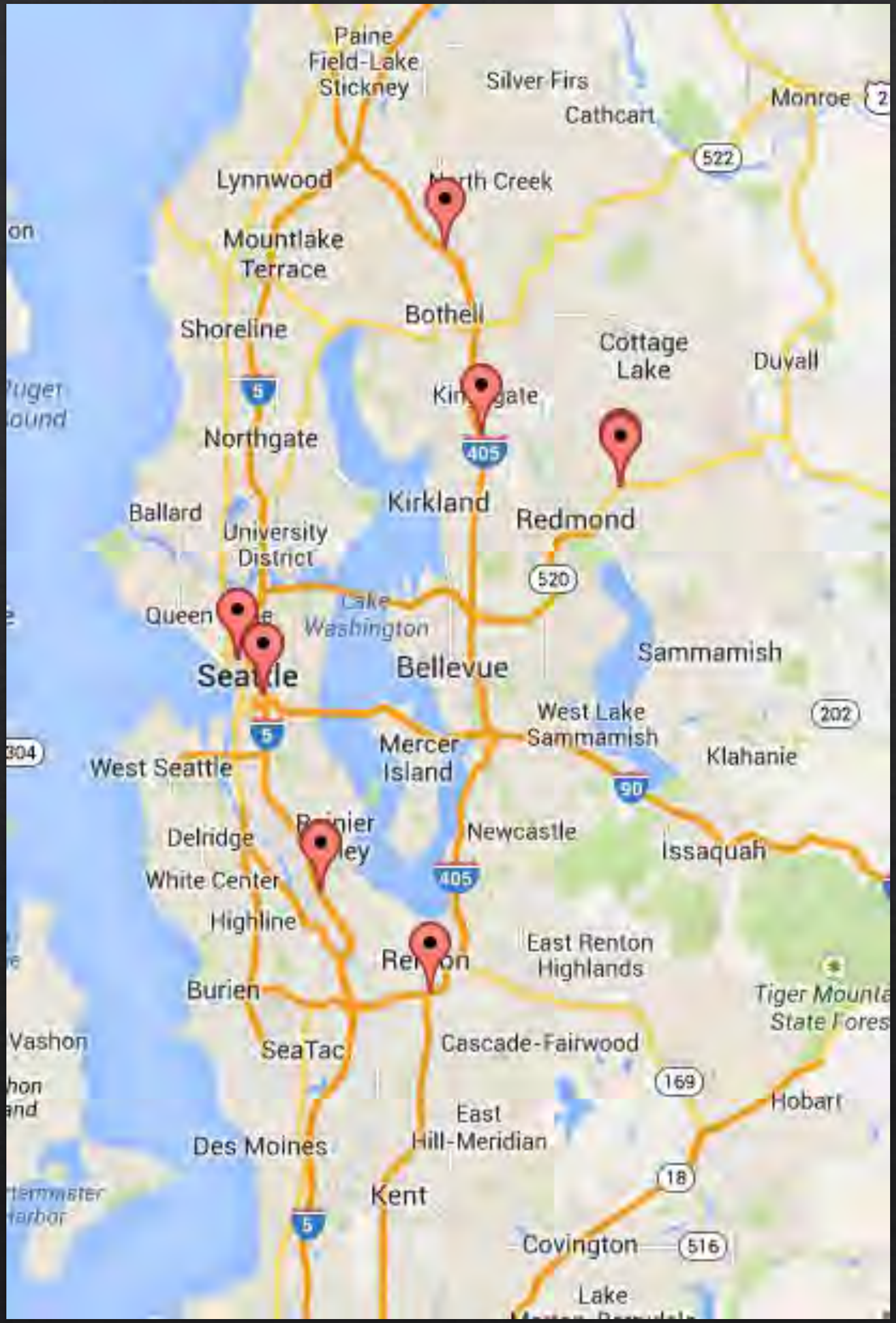
Inquiry Scan

Areas on Seattle area highways showing recurrent inquiry activity

Transmissions from 9E:8B:33:16

Locations match expected spots for route choice tracking, also DoT traffic cameras.

Notably absent: SR520 bridge, home of known Bluetooth speed sensors.



1	No.	Time	LAP	First	First Time	Last	Last Time	Spread(sec)	Spread(hours)	Latitude	Longitude	Lat(rad)	Long(rad)	Havarsine Distance(km)
349	38059	2829.617	00:00:00_14:f3:d4	1	2829.616829	0				47.7864	-122.1972	0.8340	-2.1327	
350	38233	2838.294	00:00:00_14:f3:d4	0		1	2838.294278	8.677449		47.7876	-122.1995	0.8341	-2.1328	0.216
351	103557	11877.86	00:00:00_14:fb:89	1	11877.85807	0				47.6074	-122.3333	0.8309	-2.1351	
352	103575	11884.37	00:00:00_14:fb:89	0		1	11884.37097	6.5129		47.6074	-122.3333	0.8309	-2.1351	0.000
353	101315	9326.03	00:00:00_15:14:64	1	9326.030026	0				47.6147	-122.3377	0.8310	-2.1352	
354	101336	9346.077	00:00:00_15:14:64	0		1	9346.076569	20.046543		47.6142	-122.3376	0.8310	-2.1352	0.053
355	41018	3018.286	00:00:00_15:17:d4	1	3018.285857	0				47.8177	-122.2388	0.8346	-2.1335	
356	93514	7770.216	00:00:00_15:17:d4	0		1	7770.216487	4751.93063	1.319980731	47.8182	-122.2395	0.8346	-2.1335	0.074
357	93100	7603.169	00:00:00_15:29:5e	1	7603.169205	0				47.7899	-122.2043	0.8341	-2.1329	
358	93104	7604.199	00:00:00_15:29:5e	0		1	7604.199325	1.03012		47.7901	-122.2046	0.8341	-2.1329	0.026
359	142445	13912.79	00:00:00_15:35:f2	1	13912.79191	0				47.5286	-122.1976	0.8295	-2.1328	
360	142564	13916.94	00:00:00_15:35:f2	0		1	13916.93984	4.14793		47.5290	-122.1977	0.8295	-2.1328	0.047
361	139378	13764.26	00:00:00_15:41:43	1	13764.26078	0				47.5074	-122.1978	0.8292	-2.1328	
362	139398	13765.24	00:00:00_15:41:43	0		1	13765.2432	0.98242		47.5075	-122.1978	0.8292	-2.1328	0.009
363	144421	14001.56	00:00:00_15:48:91	1	14001.55717	0				47.5357	-122.1953	0.8297	-2.1327	
364	144487	14004.32	00:00:00_15:48:91	0		1	14004.31746	2.76029		47.5359	-122.1952	0.8297	-2.1327	0.020
365	93745	7808.423	00:00:00_15:4a:0c	1	7808.422599	0				47.8235	-122.2506	0.8347	-2.1337	
366	94065	7890.994	00:00:00_15:4a:0c	0		1	7890.994015	82.571416		47.8321	-122.2631	0.8348	-2.1339	1.338
367	83903	6473.47	00:00:00_15:4c:37	1	6473.469534	0				47.7559	-122.1655	0.8335	-2.1322	
368	84392	6500.418	00:00:00_15:4c:37	0		1	6500.418111	26.948577		47.7556	-122.1655	0.8335	-2.1322	0.032
369	132331	13446.47	00:00:00_15:5b:4f	1	13446.47122	0				47.4818	-122.1986	0.8287	-2.1328	
370	132438	13450.19	00:00:00_15:5b:4f	0		1	13450.19227	3.72105		47.4821	-122.1983	0.8287	-2.1328	0.045
371	100869	9154.798	00:00:00_15:6d:e1	1	9154.797508	0				47.6188	-122.3344	0.8311	-2.1351	
372	100876	9156.076	00:00:00_15:6d:e1	0		1	9156.075585	1.278077		47.6188	-122.3344	0.8311	-2.1351	0.000
373	13422	1443.293	00:00:00_15:6e:fb	1	1443.293386	0				47.6991	-122.0932	0.8325	-2.1309	
374	13472	1445.45	00:00:00_15:6e:fb	0		1	1445.449943	2.156557		47.6988	-122.0933	0.8325	-2.1309	0.039
375	77444	5975.281	00:00:00_15:9a:79	1	5975.280565	0				47.6516	-122.1870	0.8317	-2.1326	
376	77447	5975.64	00:00:00_15:9a:79	0		1	5975.639874	0.359309		47.6516	-122.1870	0.8317	-2.1326	0.000
377	154826	14564.65	00:00:00_15:9f:97	1	14564.64898	0				47.6300	-122.1663	0.8313	-2.1322	
378	154827	14564.66	00:00:00_15:9f:97	0		1	14564.65868	0.0097		47.6300	-122.1663	0.8313	-2.1322	0.000

Passive Sniffing

Multiple areas where an address was found for <700M on each pass

Approximates the area covered by a Class 1 Bluetooth transceiver

Locations were a subset of inquiry scan locations!

Notably absent: SR520 bridge, home of known Bluetooth speed sensors

Countermeasures

- Turn off your phone!
  - Harder to turn off the Bluetooth transceivers in your car
  - Use Ubertooth to check your own signals
- Spoof other addresses
  - Find addresses with kismet, ubertooth-scan, or hcitool-scan
    - ubertooth-scan can find nondiscoverable master devices
    - ubertooth-rx can find UAP given LAP
  - Install bluesmash-tools (included in BackTrack and Kali)
  - `$ ./bdaddr -i hci0 00:01:02:03:04:05`
  - Can duplicate multiple addresses with multiple transceivers

Questions?