Digital Terrestrial Tracking: The Future of Surveillance

Glenn Wilkinson
SensePost
@glennzw

glenn@sensepost.com

ABSTRACT

In this paper, the terms *Digital Terrestrial Tracking* (DTT) and *Digital Terrestrial Footprint* (DTF) are introduced. The DTF defines the uniquely identifiable signature of wireless signals emitted by a device or collection of devices that an individual carries on their person in the physical world. These signals can reveal a device's history at a location and point in time, and potentially disclose details about the owner. Interrogation or interaction with the device may reveal further details.

The DTF positions itself between an individual's physical world footprint (their unique personal attributes), and their online footprint (defined by their unique online persona). Physical world tracking would involve following a person based on what they look or sound like; online tracking would involve tracking a person online activity based on their unique online signature (cookies, IP addresses, social media accounts); and digital terrestrial tracking involves tracking a person in the real world based on a unique signature emitted by devices on their person.

The goal of the research conducted and discussed in this paper was to build a mass data collection and correlation framework based on information leaked from the wireless devices that people carry. The framework should be able to identify, track, and profile people by passively collected wireless information from devices, and collect information that is more verbose by optionally interrogating devices.

The result is a tool, named Snoopy, written in Python, capable of operating in a distributed manner, in harsh environments on affordable off the shelf (OTS) hardware. Snoopy is able to draw specific and high level conclusions about individuals based on their digital wireless signals.

The framework has been extensively tested in busy public areas (such as conferences, airports, hotels, etc.) and validated our hypothesis that such tracking was possible. Analysis performed against the collected data revealed interesting insights and trends, which will be discussed in the results section of this paper.

Categories and Subject Descriptors

D.4.6 [Software]: Security and Protection – *Invasive Software, Verification.*

General Terms

Algorithms, Measurement, Reliability, Experimentation, Security.

Keywords

Wireless, 802.11, 802.15, Wi-Fi, Bluetooth, GSM, ZigBee, NFC, RFID, fingerprint, link-analysis, UAV, single board computer, privacy.

1. INTRODUCTION

Mobile electronic devices such as smart phones, laptops, tablets, wearable computers (e.g. smart watches, fitness bracelets, etc.) emit wireless signals, even whilst not in active use. If the signals emitted by one or more of these devices are unique, the device (and therefore the owner) can be identified as being in a certain location at a certain time. The signals may also reveal personal information about the owner, or upon interrogation or interaction divulge such information.

An example of such signals is the 802.11 wireless *probe-request* that is broadcast from mobile phones and other portable Wi-Fi enabled devices. These signals include a unique MAC address of the device, and the name (SSID) of the wireless network being searched for. The SSID may be able to be geo-located, or simple link-analysis could be conducted by identifying different devices searching for the same SSIDs, thus revealing secondary and even tertiary relationships (e.g. a spouse, or business partner).

Active interaction with devices may also be possible. For example, due to a lack of verification with WEP or OPEN Wi-Fi networks it is possible to respond to arbitrary probe requests from client devices with a *beacon*, thereby impersonating the desired access point, and intercepting network traffic from client devices. Similar techniques work with GSM, by detecting the unique IMSI (international mobile subscriber identity) of a mobile phone, with the option of operating a personal cellular tower (small cell) to intercept data. Other examples include detecting the MAC address and device name via Bluetooth; detecting the device ID with RFID; reading data from an NFC device; or detecting the device number of an ANT fitness device. Furthermore, if carrying multiple devices the cloud of device signals may provide a unique identity even if individual devices do not.

The Snoopy tool was originally released as a proof of concept for detecting, tracking, and interrogating client Wi-Fi devices [1]. Since then, it has been re-written to be modular with a technology agnostic framework able to collect signals emitted from arbitrary technologies as defined by its plugins. Current plugins include Wi-Fi, Bluetooth, ZigBee, GSM, NFC, and RFID. The framework is designed to run unattended for prolonged periods synchronizing data to a central server (via 3G modem, Xbee, or Wi-Fi). It has several modes of operation, including an aerial surveillance and tracking mode.

Visual data exploration from a central server reveals enlightening patterns-of-life of individuals, both at the macro and micro levels. This will be discussed in the results section of the paper.

Work has been done in this area by academics, private corporations, and military institutions 2-13], but at the time of writing Snoopy is the only open source technology agnostic DTF tracking and profiling framework that we know of.

2. PROBLEM STATEMENT

The goal of this research was to build a framework to detect a user's *DTF*, demonstrate how the user could be tracked via it, and what link analysis could be conducted against data in both a passive and active manner. Data collection should be in a distributed manner allowing for the deployment of low cost *sensors* (later named *drones*) over an area with data synchronised between *sensors* and *servers*. A degree of data protection was a further requirement; such that a discovered sensor would not reveal captured information to a third party.

Data exploration should be possible in a graphical manner, in such a way that operators can query data to visually discover patterns.

Finally, the devised solution should be based upon open source software and off-the-shelf hardware.

3. BACKGROUND

In this section, we discuss several wireless technologies, their common deployment in consumer devices, and examine the DTF they may create for the user.

3.1 Wireless Technology

The ever decreasing cost of production, increase in computing power, and desire for convenience is prompting a growing number of wireless enabled devices to be carried by consumers. This includes, but is not limited to; Wi-Fi/GSM/Bluetooth enabled smart phones, NFC bank cards, RFID biometric passports, Wi-Fi enabled pacemakers, low power Bluetooth tagged 'Internet of Things' items. These technologies may disclose unique identifiers, or personal information concerning the owner.

3.1.1 Wi-Fi

Wi-Fi (or more accurately the group of IEEE wireless 802.11 specifications) provides wireless access to networks at Open Systems Interconnection (OSI) model layers one (radio wave modulation) and two (primarily media access control) of the OSI network stack, and is indistinguishable from a wired (Ethernet) connection to the user or software on layers three and up. A Wi-Fi network configured in infrastructure mode (i.e. a Basic Service Set (BSS)) describes a client device as a *station,* which connects to an *access point* (AP). Ad-hoc networks are also possible between clients, but not relevant for our discussion. A typical AP has a range of 35 meters indoor, and 100 meters outdoors.

The Wi-Fi standard defines three high-level wireless *frame types*: data, management, and control. Data packets encapsulate higher-level data (e.g. IP packets); management packets control the management of the network (e.g. devices connecting to an access point); control packets mediate access to the shared medium, such as collision detection and back-off (in a similar way to Ethernet). frame types have different subtypes. For example, the *beacon*, *probe-request*, and d*eauthentication* frames are examples of management frame subtypes, while Request to Send (RTS) and Clear to Send (CTS) are examples of control frame subtypes.

Wi-Fi packets typically have three addresses associated with them: a source address, a destination address and a *Basic Service Set ID* (BSSID) address (the latter uniquely identifies an AP). These three packet types define the origin of a packet, where it is going, and which AP it should use to get there.

Security in Wi-Fi networks is broadly defined by three possibly options: Open, WEP encrypted, WPA encrypted or 802.11i (aka WPA/2) encrypted. Open networks require no passphrase to connect, whilst WEP and WPA and WPA/2 networks do. Fundamental security concerns were discovered in the WEP protocol, and WPA was designed to rectify them. WPA2 further enhanced the security.

Stations (client devices) associate to an AP by sending a *probe-request,* which includes the station's MAC address and the SSID of the desired network (alternatively, an AP may send out a *beacon* packet first). The AP responds with a probe response, after which security options and transfer rates are negotiated. Finally, the authentication process occurs (which will be dependent on the type of encryption in use, if any). If encryption is in use, only after this process will any traffic actually be encrypted (therefore it is trivial to intercept and/or spoof any of the association steps).

3.1.2 Bluetooth

Bluetooth operates in the 2.4GHz range, and is designed for transmitting data over short distances between mobile devices, and for creating Personal Area Networks (PANs).

A Bluetooth network consists of one master device, and up to seven slave devices. A slave may become a master by making a request to the current master. The master can only communicate with one slave at a time, but can rapidly change which device it is communicating with in a round-robin fashion.

Bluetooth low energy (BLE) is a subset of Bluetooth 4.0 with a new protocol stack. It is designed to have very low power consumption and run off a coin cell.

Bluetooth stacks are increasingly incorporated into smart phones, wearable technology, and a range of other low power applications. Examples include; the *FitBit* bracelet, which collects information such as sleep patterns and daily activity syncing results to a smart phone or computer via BLE; the *PebbleBee,* a personal item tagging solution which allows a user to find possessions via BLE (and allow other uses to locate them if lost); and *iBeacon*, an indoor positioning system that Apple calls "a new class of low-powered, low-cost transmitters that can notify nearby iOS 7 devices of their presence." [16].

3.1.3 ANT

ANT is a wireless sensor network technology utilizing the 2.4GHz spectrum. Unlike Bluetooth, which was designed for small star networks, ANT was designed for large-scale sensor network topologies (whilst still requiring low power at each node). It has been primarily targeted at recreation and sports, especially cycling performance monitoring. The devices are typically embedded into equipment such as cadence meters, speed monitor, and heart rate monitors. The collected data is typically relayed back to a smart phone or computer for analysis.

3.1.4 ZigBee (Wireless Sensor Networks)

ZigBee is a specification for a communication protocol to create wireless sensor networks built from small, lower power radios. Although devices are low power, transmit range can be very high (up to several kilometers). It is designed for scenarios that require a low data rate, prolonged battery life, and secure communication. Common applications include wireless meters, light switches and traffic management systems. It is intended to be cheaper and simpler than Wi-Fi or Bluetooth.

3.1.5 Cellular (GSM / UMTS)

The GSM (Global System for Mobile Communication) and Universal Mobile Telecommunications System (UMTS) are standards to describe protocols for second (2G) and third (3G) cellular networks.

When a mobile phone is turned on, it registers with the carrier communicating a TMSI (temporary mobile subscriber identity) with the nearest tower, as well as the IMSI (international mobile subscriber identity). The TMSI value rotates on tower switching, but the IMSI is constant for a device. Devices will also periodically update their location with the network (the familiar "tzzz" sound heard when a phone is near a radio). How often the location update occurs varies between country and provider.

A discussion of the security measures and algorithms in use in cellular networks are beyond the scope of our discussion, but the

IMSI catchers have demonstrated the ease with which a cellular network can be impersonated. With such a device, it is possible to determine the uniquely identifying IMSIs of all users registering with it, or to convince handsets to communicate via an attacker controlled personal cell tower (small cell). In certain conditions, it is even possible to actively signal users' handsets to instruct them to discontinue use encryption.

3.1.6 NFC (Near Field Communication)
NFC is a set of standards for establishing radio communication, and operates in the 13.56 MHz range. Communication involves an initiator and a responder, whereby the initiator generates an RF field to power a passive target. Targets may therefore be very simple devices, such as a tag, key fob, or card. The tags can store personal data such as credit card information, passport numbers, PINs, loyalty card numbers, amongst other information.

Unlike Bluetooth, NFC operates at slower speeds but does not require pairing and consumes far less power.

Whilst NFC has a theoretical maximum distance of 20cm, it has been demonstrated that with the right hardware NFC devices can be detected up to several meters away.

3.1.7 RFID (Radio Frequency Identification Tags)
RFID tags are devices that have unique identification information that can be read via an external reader. Some tags are powered by and read by magnetic fields (via electromagnetic induction), while others use a battery or collect energy from the interrogating electromagnet field and then passively emit microwaves or UHF radio waves. There are several frequencies available, with the discerning characteristic being the range and data transmission rate (from 120kHz/10cm to 3-10GHz/200m).

The tags are used in a wide array of applications, from access control; to tracking of goods; to contactless payment; to machine-readable identify documents.

Table 1 - Wireless technologies summary

Devices	Range	Unique ID	Profile Info
Wi-Fi	±100m	MAC address	SSID, rogue AP traffic interception
Bluetooth	±50m	MAC address	Device name
ANT	±50m	Device number	Device name, statistics
ZigBee	1m to 80kms	Source address	Various
GSM	35 kms	IMSI, TMSI	Femtocell traffic interception
NFC	±10cm	Various data	Personal information (e.g. passport number)
RFID	10cm to 200m	Unique ID	N/A

4. RELATED WORK

4.1 Commercial
The ability to track and profile consumers is of great relevance to the retail world. Several companies give the ability to track consumers both in store, between stores, and out of stores. These metrics are often combined with other data sources, such as credit card data, loyalty schemes, and stores cameras. RetailNext appears to be the most advanced of current offerings, giving the ability to track:

1. Video cameras
2. Wi-Fi and Bluetooth
3. Cellular
4. Guest W-Fi
5. Point-of-sales systems
6. Staffing systems
7. Promotional calendars
8. Payment cards

Other private companies such as EUCLID and Path Intelligence offer similar services.

Some solutions are opt-in and reward the user by on device applications (for example, with iBeacon) whilst others are opt-out and passively observe.

4.2 Military
There are numerous offerings in the military space for technologies that track persons, and attempt to extract data from their devices. Two such examples are the Israeli companies Netline Communications Technologies, and Verint Systems. Both over solutions to track, intercept, disrupt, and profile individuals on small, medium, and large scale.

4.3 Academic
The Snoopy proof-of-concept was released in 2012 and demonstrated the ability to track and profile individuals based on the W-Fi signal their smart phones emitted. Insights gained included:

1. Visited locations based on geo-locating SSIDs
2. Interpersonal relationships based on common SSIDs
3. Interpersonal relationships based on SSID country
4. Personal details extracted from intercepted traffic

CreepyDOL is a tool that performs some similar tasks, but only syncs data via Wi-Fi and uses a game engine for data exploration [9]. Mathieu Cunche has released numerous papers and tools exploring this arena [3-7].

5. SNOOPY
In this section we present and discuss the Snoopy framework. Snoopy is a distributed, sensor, data collection, interception, analysis, and visualization framework. It is written in a modular format, allowing for the collection of arbitrary data from various sources via the writing of simple Python plugins. For example, adding a Greentooth plugin for a newly developed wireless communication protocol would require minimal work, providing that libraries exist to interact with the technology.

5.1 Architecture
The framework consists of client devices, known as *drones* and optional server devices. Each Snoopy instance can run multiple plugins simultaneously. Each plugin collects data, which is queried by the main Snoopy process and written to a local database. Snoopy can sync data between *drones* and a server, or

drones can pull replicas of data from a server. Each Snoopy instance can run different plugins appropriate for its position in the greater picture.

Figure 1 illustrates one possible setup, whereby three *drones* collect data, and sync to two separate servers. One syncs over 3G, the other two over Xbee. The second server syncs its data to a third server. Finally, a client laptop pulls data from the first and third servers, and runs data exploration tools to examine the data. The solid lines arrow lines denote a push; the dashed arrow lines denoted a pull.

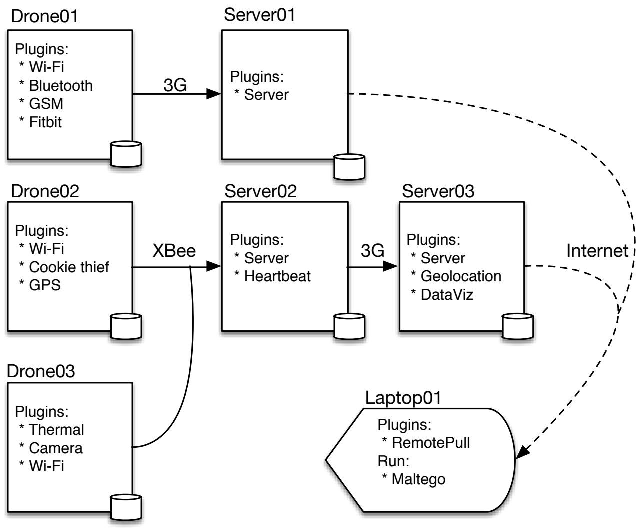

Figure 1 - Architecture Diagram

5.2 Data Collection

As mentioned in Section 5.1 the Snoopy framework can run multiple plugins simultaneously. For example, collecting data via W-Fi, Bluetooth, GSM, and GPS. It is trivial to add plugins for any device for which Python code can be written for. See Table 2 for a list of currently available plugins.

Table 2 - Existing plugins

Name	Description
c80211	Will monitor a supplied network interface (which may be automatically put into monitor mode). A series of sub-plugins exist, such that each captured packet is passed to each sub-plugin. The existing sub-plugins perform: • Note client Wi-Fi proximity • Extract vendors from observed MAC addresses • Note Access Points • Collect (actively or passively) WPA handshakes • Extract session cookies from browsed websites • Extract Apple GUID numbers.
bluetooth	Discovers Bluetooth devices.
gpsd	Queries gpsd server for GPS co-ordinates.
hearbeat	Returns a hearbeat every 60 seconds.
local_sync	Pull database from remote server and sync it into the local one.
mitmproxy	Runs a man-in-the-middle proxy server capable of modifying traffic or saving certain data (e.g. credentials).
rogueAP	Creates a rogue Wi-Fi access point, with the option to behave promiscuously.
server	Runs a server - allowing local data to be synchronized remotely.
sysinfo	Retrieves system information, every 30 minutes.
wigle	Looks up SSID locations via Wigle (from the ssid table).
gsm	Identify mobile phones based on unique identifiers such as IMSEI and TMSEI.
fitTech	Identify fitness apparel, such as FitBit.
ant	Identify ANT devices (used by cyclists).
NFC	Identify NFC devices.
RFID	Identify RFID devices.
sdr	Identify devices by using Software Defined Radio. E.g. garage doors, care remotes, airplane transponders.

The snippet below illustrates basic usage, with a single plugin (c80211). The '-d' flag specifies the arbitrary name of the *drone*, and the '-l' flag specifies the arbitrary location name.

```
# snoopy -m c80211:iface=mon0 -d myDrone -l heathrow
[+] Starting Snoopy with plugins: c80211
[+] Capturing local only. Saving to 'sqlite:///snoopy.db'
[+] Waiting for plugin 'c80211' to indicate it's ready
[+] Starting sniffing on interface 'mon0'
[+] Plugin 'c80211' has indicated it's ready.
[+] Done loading plugins, running...
```

By default, data will be saved to a SQLite file (*snoopy.db*) in the working directory. This can be overridden with the --*dbms* option, as demonstrated below.

```
# snoopy --dbms mysql://glenn:secret@localhost/snoopy_db
```

The following database engines are supported: Drizzle, Firebird, Microsoft SQL Server, MySQL, Oracle, PostgreSQL, and SQLite

5.3 Data Synchronization

Data may be synchronized from *drone* to server (push), or server to drone (pull).

```
Server
# snoopy_auth --create myDrone01
[+] Creating new Snoopy server sync account
[+] Key for 'myDrone01' is 'GWWVF'
# snoopy --plugin server
[+] Running webserver on '0.0.0.0:9001'
[+] Plugin server caught data for 2 tables.
```

```
Drone
# snoopy -m c80211:iface=mon0 -d myDrone -l heathrow -k
GWWVF -s http://<server>:9001/
[+] Snoopy successfully sunc 2 elements over 2 tables.
```

The *pull* operation acquires a replica of all data on the server. This option is illustrated below.

```
Drone
# snoopy -m local_sync:server_url=http://<server>:9001/ -
d myDrone -m -k GWWVF

[+] Plugin local_sync pulled 138 records from remote
server.
```

5.4 Data Visualization & Exploration

Data is stored in a database, and may be queried in several ways. The most trivial is to use the appropriate command line tool (for example, *sqlite3* if saving in SQLite format). The preferred method is with Maltego, a data visualization and graphing engine. Entities are populated onto a canvas, and *transforms* can be run against them, to query data. Transforms may either be run locally, or via a remote Transform Distribution Server (TDS) server. The benefit of running queries against a TDS server is that the analyst laptop does not require any additional libraries or tools, and only requires the Maltego software. This is depicted in Figure 2.

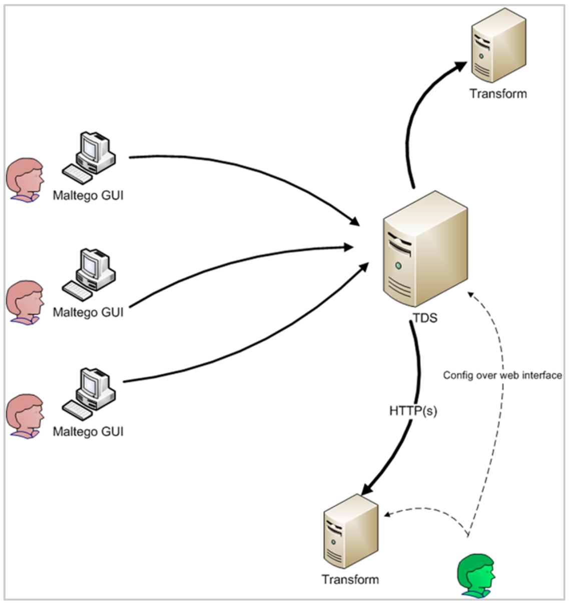

Figure 2 - Maltego TDS

A simple web interface also allows exploration of data. This includes plotting of data onto a map, by both time series data, and static.

5.5 Modes of Operation

The *drone* devices have five broad deployment options defined by the environment they are to be used in. These are *Sentry, Mobile, Aerial, Ground Vehicle, and Telescope.*

5.5.1 Sentry

The Sentry configuration consists of a more ruggedized enclosure with a long life battery and/or a solar cell. It is intended to be deployed for medium to long term in an environment exposed to the elements (for example, under a park bench, on a telegraph pole, or in a tree).

5.5.2 Mobile

Snoopy has been tested on two consumer mobile platforms, namely the Google Nexus 7 and the Nokia N900. Running the software from such a device would lower suspicion, due to the perceived harmless nature of the device.

The GPS plugin collects positional data to compliment this mode of operation.

5.5.3 Aerial

A Snoopy *drone* running on a *BeagleBone* single board computer has been successfully deployed on an Autonomous Aerial Vehicle (UAV), thus enabling beyond line-of-sight data collection. It is possible to search for and follow a known DTF with this mode by instructing the on-board flight controller. See Section 5.7 for a further discussion.

The GPS plugin collects positional data to compliment this mode of operation.

5.5.4 Ground Vehicle

Similarly to the above aerial scenario, a *drone* has been successfully deployed in a motor vehicle and bicycle.

The GPS plugin collects positional data to compliment this mode of operation

5.5.5 Telescope

The *telescope* configuration denotes using a high gain / high sensitivity antenna from a distant vantage point. It is intended to 'sweep' an area to collect DTFs, or to discover the general location of a known DTF (thereafter allowing mobile and/or aerial units to discover the precise location of the DTF).

5.6 Search Patterns

Figure 3 illustrates a spiral search pattern. At time *t=0* the *drone* begins its search for a known DTF. It ascends to a fixed altitude, and performs an outward spiralling pattern until obtaining the desired DTF. At this point, the unit may simply take a photograph and return, or maintain a follow pattern (see Section 5.7). Multiple *drones* can combine this strategy over a large area, with each having a designated area to search.

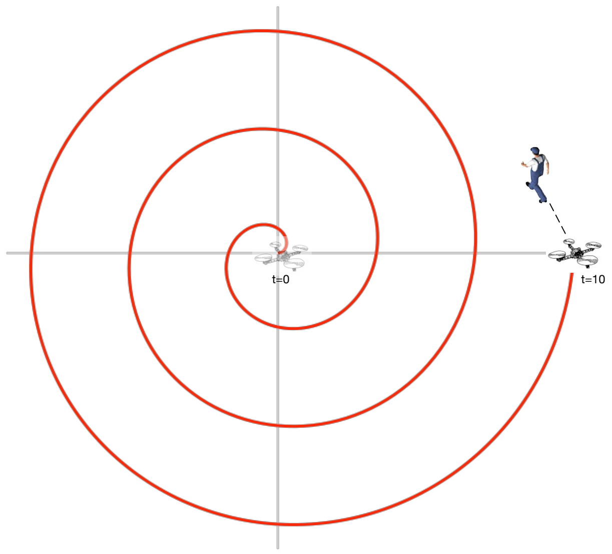

Figure 3 - Spiral Search Pattern for Known DTF

5.7 Trilateration & Follow

Whilst *triangulation* uses angles to locate points, *trilateration* uses lateral distances (this is the technique used with GPS satellites). If three positions (P1, P2, P3) are known, as well as the distance from each point (r1, r2, r3), then the formed overlapping circles can estimate a position relative to the three points. This can lend itself to the situation whereby three drones (or one drone in motion, with a reading from three positions) can determine the position of a DTF relative to their own GPS co-ordinates (see Figure 4). The distance metric can be calculated by the dB signal strength of a DTF. The *free-space path loss* (FSPL) characterizes how a wireless signal degrades over distance (following an inverse square law). This is illustrated by the following equation:

$$FSPL(dB) = 20\log_{10}(d) + 20\log_{10}(f) + 92.45$$

The constant value of 92.45 will vary depending on the units chosen for other measurements. In the standard example above GHz and kilometers are used for frequency and distance, respectively. Other authors [27] recommend a value of -27.55, which treats frequency in MHz and distance in meters. The equation can be used to solve for *d* in Python as follows:

```
def calcDistance(db, freq):
    exp = (27.55 - (20 * math.log10(freq) + db) / 20.0
    return math.pow(10.0, exp)
```

This technique will work significantly better for tracking outdoor signals due to lack of obstruction, thus simplifying the free space calculations.

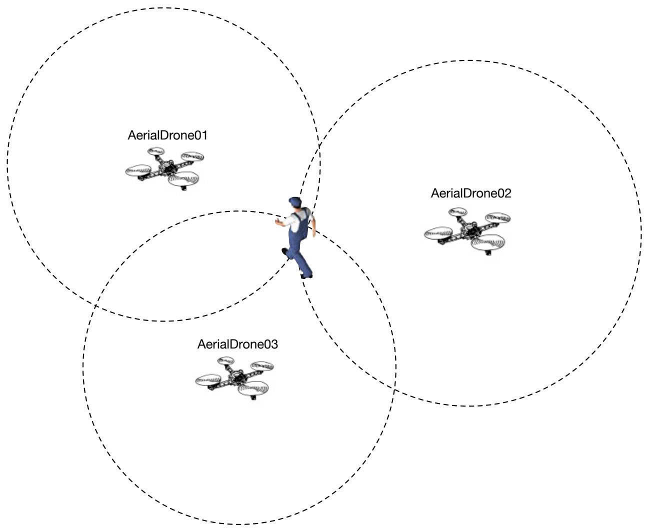

Figure 4 – Trilateration of Known DTF

5.8 Hardware

Snoopy will run on any modern Linux based device, with the appropriate peripherals for the desired plugins. It has been tested on a laptop, the Nokia N900 (running Maemo Linux), the Nexus 7 tablet, and the BeagleBone Black single board computer. The later is the preferred method of deployment. Table 3 lists several hardware peripherals for collecting data of several technologies.

Table 3 - Sample peripheral devices

Technology	Hardware
Wi-Fi	AWUS 036H
Bluetooth	Ubertooth
ZigBee	Digi Xbee
GSM	RTL2832U SDR
RFID	RFidler
NFC	ACR122U

6. RESULTS

The framework has been tested in numerous environments, two of which will be discussed here. The first involved a 14-hour deployment at a busy train station in London. The second involved collecting data at numerous security conferences over the period of one year. At the time of writing, the Wi-Fi plugin generates the most comprehensive DTF, and thus results from this plugin are discussed in this section.

6.1 Train Station

The framework was run on a laptop whilst sitting at King's Cross railway station in London on the 23rd August 2012. During the 14-hour experiment 42,480 unique devices were observed. Figure 5 illustrates the distribution of observed devices, with clear peaks observed during the morning, lunch, and evening periods. Figure 6 illustrates the proportion of this number per manufacturer (as determined by devices' MAC addresses).

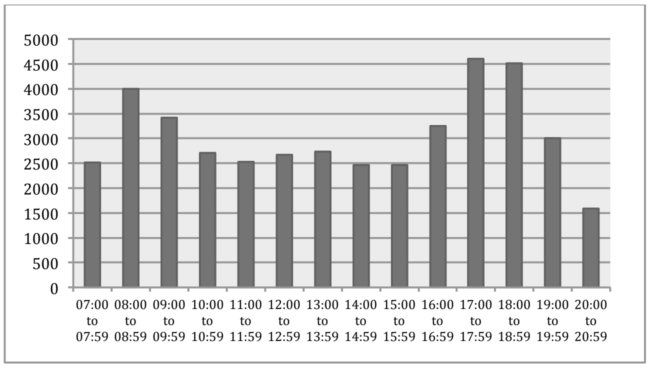

Figure 5 – Devices at Kings Cross station

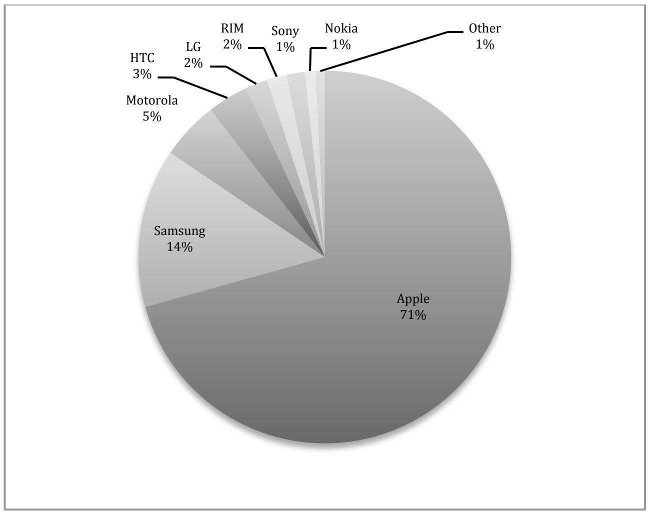

Figure 6 - Proportions by manufacturer, Kings Cross station

6.2 Security Conferences

The framework was run at numerous security conferences over a two-year period. Table 4 lists the number of devices that were observed at the events, along with the approximate number of attendees.

Table 4 - Conference attendee devices

Conference	Year	Unique Devices	± Attendees
Black Hat Vegas	2012	4,778	6500
ITWeb		1,106	400
44Con		969	350
Black Hat EU	2013	681	607
Securi-Tay		375	100
BSides London		208	474
Hackito-E-Sum		309	400
CERT Poland		598	500
Zero Nights		507	600
Black Hat Brazil		719	500

Figure 7 depicts devices observed at the Black Hat Brazil conference that were also observed at other conferences.

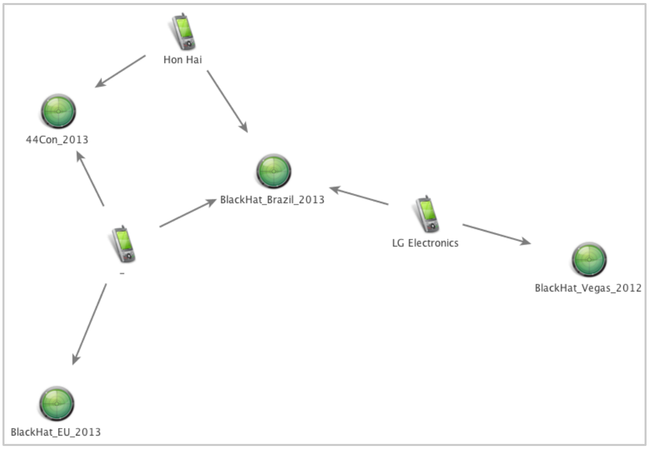

Figure 7 - Repeat visitors to Black Hat Brazil 2013

Figure 8 illustrates client device previous network SSIDs geo-located from the Black Hat Brazil 2013 security conference.

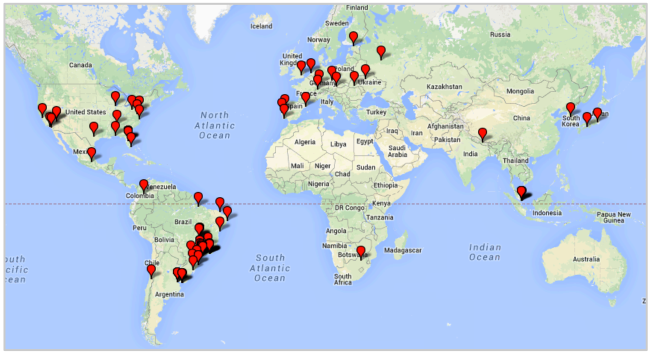

Figure 8 - Visitors to Black Hat Brazil 2013

6.3 Link Analysis

The Maltego platform allows for the easy creation of arbitrary data querying constructs known as *transforms*, which are run against *entities*. A sample of transforms from the Wi-Fi suite is presented in Table 5.

Table 5 - Wi-Fi transforms

Input Entity	Transform	Output Entities
Base of Operations	fetchDrones	Drone
Drone	fetchLocations	Location
Location	fetchClients	Client device
Client	fetchSSIDs	SSID
Client	fetchObservations	Observation
SSID	fetchSSIDLocation	Address
SSID	fetchClients	Client
Address	fetchNearbySSIDs	SSID
Client	fetchDomains	Domain
Domain	fetchCookies	Cookie
Domain	fetchFacebook	Facebook
Facebook	fetchFriends	Facebook

Furthermore, filtering can be performed by drone, location, or observation date/time. For example, it is possible to query all devices observed in Finsbury Park on the 11 March 2014 between 1pm and 1:30pm.

6.3.1 Shared Location

It is possible to query the database for devices that have some value in common. For example, searching for all mobile phones that are looking for a common SSID, but ignoring generic results (such as "Starbucks"). This can quickly disclose relationships such as work colleagues (devices looking for a SSID at a determined business address) or family members (devices looking for an SSD at a known residence). Figure 9 illustrates four devices searching for the same network *RBS-1-1111*, which was geo-located to a Royal Bank of Scotland Branch in Liverpool Street. This observation was made on a train into London.

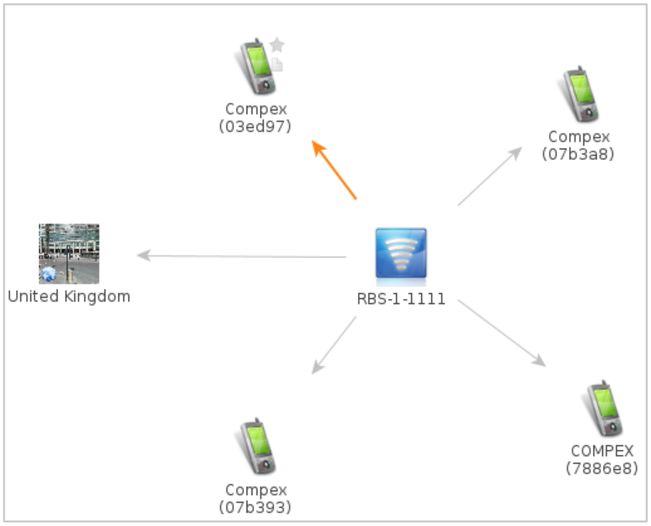

Figure 9 - RBS devices

6.3.2 Known Prior Location

It is possible to determine whether any devices are nearby have visited a predetermined location either by having previously collected data at the prior location, or by having information about the prior location that can be queried. For the later example it is possible to search for all known AP SSIDs from an address, and query the database to see if any devices within our proximity have searched for any of those SSIDs.

This is illustrated in Figure 10 in which we query all AP SSIDs within a 200m radius of the SensePost office (as determined by a photograph on the SensePost website with EXIF data). Each returned SSID is then queried to see if any observed devices have ever searched for one of those SSIDs. In this real world (but redacted) example four SensePost devices were discovered at Black Hat 2013. Each of those discovered devices could be queried further to discover personal information about SensePost employees.

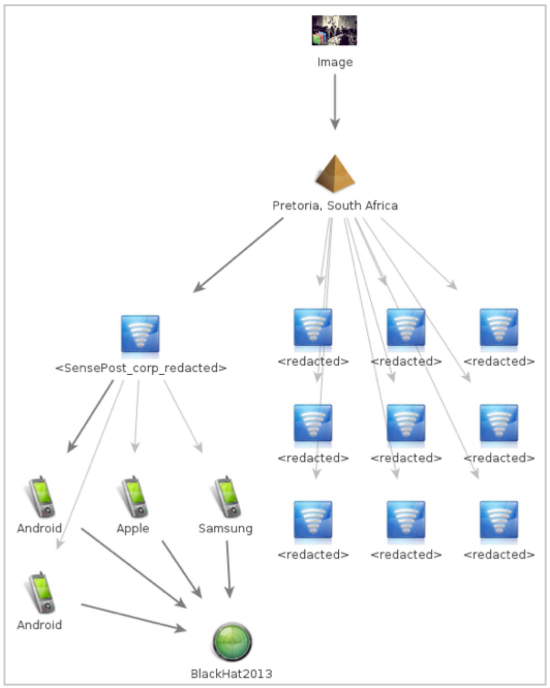

Figure 10 - Discovering SensePost devices

6.3.3 Multi-hop Analysis

Figure 11 depicts the end of a several steps of investigation. An Apple device of interest was noted to be at three locations: Starbucks, Hyde Park, and Heathrow airport. The analyst then queried the SSIDs of networks the device was looking for which revealed numerous results; each SSID was then queried to see if any other devices had been searching for any of the same networks, which revealed the Blackberry device which was also searching for the AGMC_GUEST network. This implies that both those devices have visited the same location at some point in the past (not necessarily simultaneously). This SSID is geo-located to Dubai, and an address and street view photograph is retrieved.

The same process is conducted with the Blackberry device, which reveals a third link: an HTC device searching for the common Verizon SSID. This is geo-located to San Francisco. At this point active traffic interception was conducted against all three devices, from which Facebook profiles were acquired for two, revealing common friends. This result reinforces the original hypothesis that the owners of the HTC and Blackberry know each other.

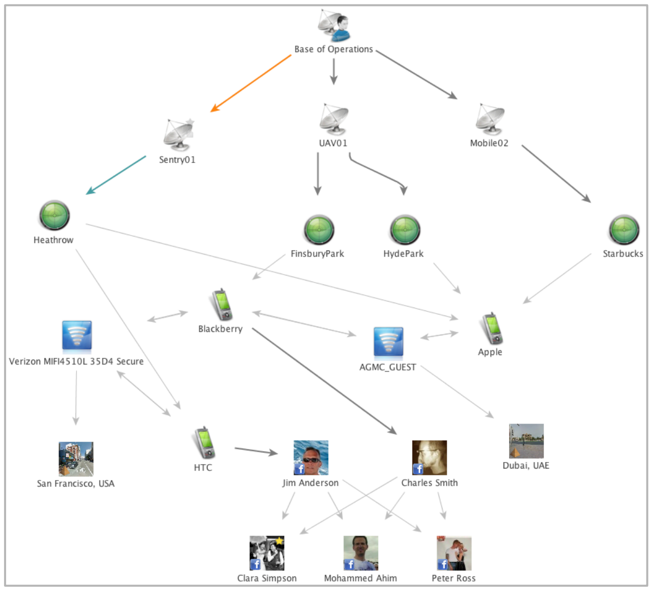

Figure 11 - Multi-hop analysis

6.4 Other Scenarios

The examples in the previous section depict common scenarios that have been encountered. Furthermore, several scenarios are presented below.

6.4.1 Finding the Spy

Assume that drones are running at numerous locations – from hotels, to airports, to rail stations. Partial intelligence is received that a person of interested will arrive during a certain week, and check into one of three hotels, and depart via one of two train stations a week later. It would be possible to look for overlap of one (or a few) devices from the locations and temporal information to isolate the target.

6.4.2 Discovering a Celebrity

If we know that a celebrity's schedule we can attempt to be within proximity of that person at numerous events, and observe the overlap of devices observed at multiple events. This process would be repeated until only one device was observed at all events that the target celebrity attended.

6.4.3 Understanding Customers

Customer tracking was discussed in Section 4.1, and is currently an area of much debate regarding privacy and invasion. It would be of great interest to a retail store to have drones deployed at numerous stores to track how long customers spend in stores, how they migrate between stores, and the effects of promotions or other external factors affect numbers.

6.4.4 Catching Criminals

Suppose an entire city is covered with drones. If a crime is committed at a location historical, records could be investigated to determine devices present near the scene of the crime.

Alternatively, suppose a riot is taking place in a city. Aerial drones could be flown overhead to collect device signatures from the rioters below, and used for future prosecution, or to discover the identities of the persons on the ground.

7. Opting Out / Defences

If a consumer does not want to be tracked based on the discussions in this paper they should take care to be aware of their own footprint. Take note of every electronic device carried on your person, and determined what wireless technologies are in use, and how unique the signals emitted are. The Snoopy framework is a useful tool for assessing one's own footprint, before those for whom you would not want to know it capture it.

8. FUTURE WORK

Additional research is planned for fixed wing aerial units due to considerably longer flight time, as well as optimizations and additional interfaces to the framework as a whole. As new wireless technologies are released, additional plugins will be added to the framework.

9. CONCLUSIONS

In this paper the unique signatures that mobile devices emit was introduced. From this initial observation a hypothesis was constructed with the intention of tracking people based on devices they carry, as well discovering personal information about them. Numerous wireless technologies were discussed, identifying their range and possible fingerprint. The Snoopy framework was then introduced as a distributed, tracking, profiling, data interception, and analysis tool. The framework is technology agnostic and can be used to track any signature for which a suitable Python plugin can be written. Result data was discussed from field experiments with the framework, and additional hypothetical scenarios were discussed. Finally, advise was given on how to be aware of one's own footprint, as well as a mention for future work.

10. ACKNOWLEDGMENTS

Thank you to SensePost for providing the time and resources to conduct this research, and to you the reader for your interest in this topic. Additionally, thank you to the team who run the WIGLE project for their insight and assistance.

11. REFERENCES

[1] Bock, J., & Lynn, M. (2007). In *Hacking Exposed Wireless*. McGraw-Hill, Inc.

[2] Barbera, M. V., Epasto, A., Mei, A., Perta, V. C., & Stefa, J. (2013, October). Signals from the crowd: uncovering social relationships through smartphone probes. In *Proceedings of the 2013 conference on Internet measurement conference* (pp. 265-276). ACM.

[3] Cheng, N., Mohapatra, P., Cunche, M., Kaafar, M. A., Boreli, R., & Krishnamurthy, S. (2012, October). Inferring user relationship from hidden information in wlans. In *MILITARY COMMUNICATIONS CONFERENCE, 2012-MILCOM 2012* (pp. 1-6). IEEE.

[4] Cunche, M., Kaafar, M. A., Chen, J., Boreli, R., & Mahanti, A. (2012, October). Why are they hiding? Study of an anonymous file sharing system. In *Satellite Telecommunications (ESTEL), 2012 IEEE First AESS European Conference on* (pp. 1-6). IEEE.

[5] Cunche, M., Kaafar, M. A., & Boreli, R. (2012, June). I know who you will meet this evening! linking wireless devices using wi-fi probe requests. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a* (pp. 1-9). IEEE.

[6] Cunche, M. (2013). I know your MAC Address: Targeted tracking of individual using Wi-Fi. *Journal of Computer Virology and Hacking Techniques*, 1-9.

[7] Cunche, M., Kaafar, M. A., & Boreli, R. (2013). Linking wireless devices using information contained in Wi-Fi probe requests. *Pervasive and Mobile Computing*.

[8] Musa, A. B. M., & Eriksson, J. (2012, November). Tracking unmodified smartphones using wi-fi monitors. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems* (pp. 281-294). ACM.

[9] O Connor, B. (2013). CreepyDOL: Cheap, Distributed Stalking. In *BlackHat 2013*.

[10] Bankston, K & Soltani, S. (2014). Tiny Constables and the Cost of Surveillance. In *The Yale Law Journal Online.*

[11] Soltani, S. (2014). Mobile Device Tracking. In *FTC Spring Privacy Series.*

[12] Netline Communications Technologies. Detection Products. http://www.netlinetech.com/products/detection.html

[13] Verint Systems. Communications Interception. http://www.verint.com/solutions/communications-cyber-intelligence/solutions/communications-interception/index

[14] Cuthbert, D & Wilkinson, G. (2012) Snoopy: Distributed tracking and profiling framework. In *44Con 2012*

[15] PebbleBee: Versatile iOS/Android Bluetooth Device http://www.pebblebee.com/.

[16] Apple Inc. (2013). Submit your iOS 7 apps today. https://developer.apple.com/ios7/

[17] WiGLE: Wireless Geographic Logging Engine. http://wigle.net/.

[18] Lee, E. (2012). NFC hacking: The easy way. In *Defcon* 20

[19] Lifchitz, R. (2012). Hacking the NFC credit cards for fun and debit. *Hackito Ergo-Paris, France, poslední aktualizace*, *13*(4).

[20] Federrath, H. (1999). Protection in mobile communications. In: *Günter Müller, Kai Rannenberg (Ed.): Multilateral Security in Communications, Addison-Wesley-Longman 1999, 349-364*

[21] Borgaonkar, R., Golde, N., & Redon, K. (2011). Femtocells: a Poisonous Needle in the Operator's Hay Stack. *Black Hat Las Vegas*.

[22] Cavallini, A. (2013). iBeacons Bible. http://meetingofideas.files.wordpress.com/2013/12/ibeacons-bible-1-0.pdf

[23] Digi XBee Radios: http://www.digi.com/xbee/

[24] Hancke, G. (2008, July). Eavesdropping attacks on high-frequency RFID tokens. In *4th Workshop on RFID Security (RFIDSec)* (pp. 100-113).

[25] Dynastream Innovations. (2013). ANT Message Protocol and Usage

[26] Ryan, M. (2013). Bluetooth: with low energy comes low security. In *7th USENIX Workshop on Offensive Technologies.*

[27] Miller, R. (2013). Wifi-based trilateration on Android. http://rvmiller.com/2013/05/part-1-wifi-based-trilateration-on-android/