# Distributed Tracking and Profiling Networks

## The good, the bad, and the downright creepy…

Brandon Sinkovic, Matthew Lysyj, Nicholas White

## Introduction

Numerous initiatives from governments around the world have sought to legalize the monitoring of citizens' Internet based communications. The average user knows little about how their devices work, and how they can be used as an extreme form of privacy invasion.

The purpose of the research being done here is to not only help illustrate this point, but to also provide several solutions to the privacy issue.

Using WiFi, the DTaPN seeks to profile each host (end user). With the information gathered, the DTaPN tries to build a model to represent the users in the area, and provide as much data about them as possible.

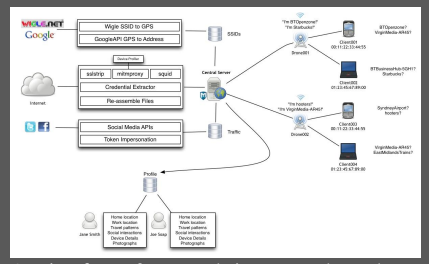
Given the RFC documentation on WiFi and client devices, the information in this documentation can be used to either exploit, or enrich the functionality of modern mobile devices. DTaPNs can track and profile users based on their location, internet traffic, history of connected routers, and even their name.

## Experiment

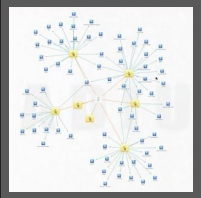
Unlike most research projects being presented, this one is not finished. The reason is that the drone is collecting data right now, YOUR data. Anyone carrying a smart phone, tablet, or WiFi –enabled device is in the data shown in front of you. All of their network traffic, location changes, and even their home addresses are being computed and stored in a database IN REAL TIME.

A large number of people leave their WiFi on, even security savvy folk. As per the RFC documentation, client devices send out 'probe requests' looking for networks that the devices have previously connected to. This feature serves as a transparent convenience to the user, allowing them to automatically, and quickly switch between wireless networks.

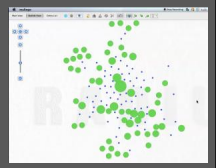
First, a drone is made using the DTaPN software. This software is highly customized for Linux devices, and can currently run on some Nokia phones, Raspberry Pi computers, and laptops. Once the drone is ready, it is signaled to begin monitoring. Each drone collects every observed probe-request, and uploads it to a central server (timestamp, client MAC, SSID, GPS coordinates, and signal strength). On the server side client observations are grouped into 'proximity sessions', and therefore we can infer was within proximity to that particular drone during that time. We now know that this device (and therefore its human) were at a certain location at a certain time. Given enough monitoring stations running over enough time, we can track devices/humans based on this information.



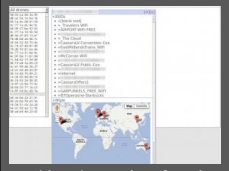

Overview of a sample DTaPN – devices contact drones who then pull information from the phones and store the information on remote servers



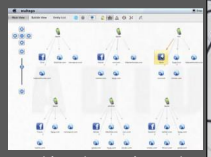
View of how a DTaPN can map the topology of a network, and help those using the drones better understand the individuals who are monitored by the drones.



Sample data of actual internet searches and websites visited by individuals unknowingly being monitored by a drone.



Tracking the previous location of a device using its historical SSIDs



Tracking the previous web viewing history of devices within drone's range

Real time tracking of a device using drone

Topology map produced by DTaPN

Topology map produced by DTaPN

Quadcopter which can be used to carry a DTaPN drone

## Results and Ethicality

Initial experiments have shown that the drones and DTaPN are capable of collecting sensitive, and sometimes extremely private information. The simplicity of the drones, and ability for them to be quickly and cheaply made are signs that they can be utilized easily, and therefore not out of reach for even the most novice of black-hat hackers.

Some legal uses for the DTaPN are collecting anonymized statistics on throughfare. One example of this is to gather statistics on peak human traffic in crowded places such as cities, subways, etc… Such statistics would allow city officials to deploy more staff, or open and close roads how they see fit.

Along with the pros, there are many borderline, and illegal activities the DTaPN makes extremely easy and easy to reach. If, for example, advertisers would decide to use the data to track the frequent location of those in the area, and those in their stores, they could gain valuable, but not consensual knowledge about all of us.

The very worst of outcomes related to DTaPNs are the fact that they make targeting and tracking individuals extremely easy. Another risk is that the drones could be used to infect users' devices, either by injection malicious web traffic, or firing exploits from the drone's server at devices.

## Further Research

With the many dangers of DTaPNs in mind, the focus of the group's further research is into preventing devices from 'falling for' this trick.

We have begun development on an application which seeks to prevent the phone or other mobile device from automatically pinging the drones.

The app is in its testing phase, and will be available within a year.

## Acknowledgements

Special thanks to COF and NSF for allowing us the opportunity to do this research.
Special thanks to Victoria Holt for her contributions to the project in both software development and mapping functions.
Special thanks to Maltego and Sensepost for the Snoopy code, and the mapping software.

```
def get_data(self):
    """Fetch data from all plugins"""
    for m in self.modules:
        multidata = m.get_data()
        if multidata:
            for rawdata in multidata:
                if rawdata is not None and rawdata:
                    tbl, data = rawdata
                    if data:
                        for i in range(len(data)):
                            if m.getName() != "server" and m.getName() != "local_sync":
                                data[i]['run_id'] = self.run_id
                        self.all_data.setdefault(tbl, []).extend(data)
                        if self.verbose > 2 and m.name != 'run_log':
                            logging.info("Plugin '%s%s%s' emitted %s%d%s new datapoints for table '%s%s%s'." %(GR,m.name,G, GR,len(data
def write_local_db(self):
    """Write local db"""
    for tbl, data in self.all_data.iteritems():
```

Sample section of Python code used to read the data from the device and store it into the database