Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Your WiFi is leaking: What do your mobile apps gossip about you?

John S. Atkinson[a,*], John E. Mitchell[a], Miguel Rio[a], George Matich[b]

[a] University College London, UK

[b] Selex ES, UK

HIGHLIGHTS

- Analysis of WiFi privacy leaks despite encryption.
- Inference of personal information from personalised mobile app usage.
- Live detection on real-world networks.
- Demonstration of a practical threat to real-world users.
- Techniques can generalise to other protocols e.g. 4G LTE.

ARTICLE INFO

Article history:
Received 31 August 2015
Received in revised form
22 May 2016
Accepted 25 May 2016
Available online xxxx

Keywords:
WiFi
Mobile apps
Privacy
Security
Data protection
Information inference

ABSTRACT

This paper describes how mobile device apps can inadvertently broadcast personal information through their use of wireless networks despite the correct use of encryption. Using a selection of personas we illustrate how app usage can be tied to personal information. Users would likely assume the confidentiality of personal information (including age, religion, sexuality and gender) when using an encrypted network. However, we demonstrate how encrypted traffic pattern analysis can allow a remote observer to infer potentially sensitive data passively and undetectably without any network credentials.

Without the ability to read encrypted WiFi traffic directly, we process the limited side-channel data available (timings and frame sizes) to enable remote app detection. These side-channel data measurements are represented as histograms and used to construct a Random Forest classifier capable of accurately identifying mobile apps from the encrypted traffic they cause. The Random Forest algorithm was able to correctly identify apps with a mean accuracy of ∼99% within the training set.

The classifier was then adapted to form the core of a detection program that could monitor multiple devices in real-time. Tests in a closed-world scenario showed 84% accuracy and demonstrated the ability to overcome the data limitations imposed by WiFi encryption. Although accuracy suffers greatly (67%) when moving to an open-world scenario, a high recall rate of 86% demonstrates that apps can unwittingly broadcast personal information openly despite using encrypted WiFi. The open-world false positive rate (38% overall, or 72% for unseen activity alone) leaves much room for improvement but the experiment demonstrates a plausible threat nevertheless.

Finally, avenues for improvement and the limitations of this approach are identified. We discuss potential applications, strategies to prevent these leaks, and consider the effort required for an observer to present a practical privacy threat to the everyday WiFi user. This paper presents and demonstrates a nuanced and difficult to solve privacy vulnerability that cannot not be mitigated without considerable changes to current- and next-generation wireless communication protocols.

1. Introduction

WiFi communications are now an everpresent part of modern society; pervading homes, business and almost everything between. The availability of WiFi and cellular data plans has led to an explosion of popularity in mobile devices (phones and tablets) and the apps that run on them. As will be discussed in Section 2, the ability to infer information about encrypted communications via side-channels has been previously established, as have the privacy implications of persistently carrying personal mobile devices and the potential security risks of particular apps and services. However, this paper demonstrates how the three in combination

[*] Corresponding author.
E-mail address: j.atkinson@ee.ucl.ac.uk (J.S. Atkinson).

http://dx.doi.org/10.1016/j.future.2016.05.030

present a perfect storm making users' private and sensitive information vulnerable. This information can be leaked to any listening party within reception range of the wireless network. The observer can operate despite WiFi encryption working exactly as designed, requires no access credentials, and can perform the analysis on commodity hardware. The technique is therefore remote, passive, undetectable and inexpensive.

To demonstrate this, 34 highly-ranked apps were chosen and the target demographics of their users identified. Network data was then collected as the apps were opened. This network activity denotes the use of a particular app. However, due to encryption only limited information is available in the form of side-channels. Interpretations of frame size and interarrival time characteristics were used to create histograms detailing the distribution of these metrics over a given time period. The distributions can then be used to differentiate between samples of encrypted network activity from different apps. These distributions were labelled appropriately and used by a Random Forest machine learning algorithm to produce a classifier that predicts app usage (or lack thereof) based on samples of encrypted network activity. The Random Forest was then converted to compiled code for speedy analysis of data in real-time. Personas were created to emulate different people, possible app choices, and then monitored in real-time to demonstrate how personal information can be leaked.

The methods used to identify apps could also be used to 'fingerprint' other activity over encrypted communications (*e.g.* VoIP, websites). However, the personal ties of mobile apps, an openly ranked market with relatively low diversity (compared to website fingerprinting in other work), and ease of collection make apps a particularly opportune and vulnerable target. Although the processes in this paper are demonstrated using standard 802.11g WiFi, the methods should generalise to other wireless communications protocols unless they are specifically designed to resist this type of analysis. Notably for mobile apps, the measurements used to perform this analysis will also be present in longer range protocols like 4G LTE in cellular phone networks.

1.1. Scenario and the WiFi environment

The scenario presented here will be familiar to all readers: a mobile device connects to a WiFi Access Point (AP) providing internet access. The apps on the device may then use this connection to communicate with certain remote internet servers. Information from these servers is used to provide the app's content or functionality. Although reliant on an internet connection, this centralised architecture minimises the storage and processing power required by the device itself. Devices can therefore be smaller, cheaper, and provide up-to-date content or backups whenever connected. Furthermore, as discussed in Section 3.2, we found that even those apps that only provide static local content may still use the internet connection to some degree when available. In our scenario the AP provides an 802.11g WiFi network utilising industry standard encryption (*i.e.* WPA2-PSK or WPA2-Enterprise with EAP) so only authorised users can access the network.

As illustrated in Fig. 1, our adversary is a remote observer attempting to infer information about the users of mobile devices connected to an AP. As the observation process is entirely passive, the observer is completely undetectable by the user or network operator. They are not necessarily malicious but have no access to the network's credentials or security keys, nor do they ever attempt to crack or discover them. As will be discussed in Section 6, the methods presented in this paper can be adapted for a variety of purposes including virtuous ones.

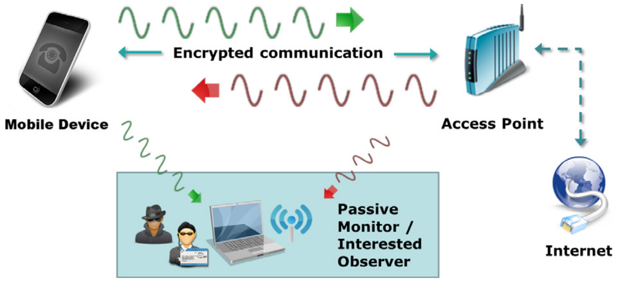

Fig. 1. Observing encrypted mobile device WiFi traffic.

1.2. Contributions

This paper details the following contributions:

1. An explanation of how sufficient information can be extracted from external observations of encrypted WiFi traffic to identify specific user activities. We describe how timing and frame size measurements over a short time period can be interpreted as histograms. The values of these histogram bins drive classification via machine learning. Despite the highly data-limited scenario this information can be used to accurately 'fingerprint' different activities from a remote, unprivileged vantage point.
2. We apply this fingerprinting technique to a selection of widely-used mobile apps and construct a Random Forest capable of accurately identifying these apps from the encrypted activity they cause. We collect on-device traffic and then convert the measurements to how it would appear from the perspective of a remote, unprivileged observer. The Random Forest algorithm was able to correctly classify apps with mean accuracy of ∼99% for this training set.
3. Due to the highly personalised nature of many mobile apps, we show that the ability to fingerprint these apps without breaking encryption can leak private information. A selection of personas demonstrate how personal information can be linked to the apps people use. The average user is unlikely to realise that such information is potentially being broadcast when they use these apps despite encryption working exactly as designed.
4. We demonstrate that the app identification process is efficient enough to allow real-time monitoring of multiple devices. Using the classifier developed from the training data, a live, remote detector program is developed and its performance in an open-world and closed-world scenario is assessed. The closed-world showed good performance (84% accuracy) and demonstrated the ability to overcome the data limitations imposed by WiFi encryption. Furthermore, the ability to adapt signatures generated from (easier to collect) on-device samples to function from an external vantage point was verified. Although accuracy suffers greatly (67%) when moving to an open-world scenario where previously unseen network activities are introduced, a high recall rate of 86% (denoting correct identification of an app providing the app was active) demonstrates the ability for apps to broadcast personal information despite WiFi encryption. The open-world false positive rate (38% overall, or 72% for unseen activity alone) leaves much room for improvement but demonstrates a plausible threat nevertheless.
5. Applications, limitations, potential avenues of improvement and strategies to prevent these WiFi leaks are identified. Finally, we assess the cost and effort required for an interested observer to present a practical privacy threat to the everyday WiFi user.

Please cite this article in press as: J.S. Atkinson, et al., Your WiFi is leaking: What do your mobile apps gossip about you?, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.05.030

1.3. *Paper outline*

The next section (Section 2) summarises the related work providing the foundation for this study: wireless network security, mobile devices and app privacy, and inference of information using only side-channels. Section 3 describes the mobile app selection process, how their activity was measured, and details the construction of the Random Forest classifier that was capable of operating from such a unprivileged vantage point. Section 4 reports the resultant classifier's accuracy on our sample data before evaluation on low-cost hardware in the real-world. Section 5 then explains the creation of personas representing real-world users and subsequent testing in closed- and open-world scenarios to demonstrate how app usage detection can betray sensitive information. The implications and practicality of these findings are then discussed in Section 6 before the paper is concluded in Section 7.

2. Research background

2.1. *Wireless networks and security*

WiFi is now almost ubiquitous in the developed world. Its pervasive deployment saturates residential, industrial, business and government buildings alike. Wireless networks are so prevalent that utilising broadcasts of nearby APs to pinpoint the location of the receiver is a common feature of modern mobile devices [1,2]. Although broadcast over a wide area, encryption is used to keep WiFi communications confidential. However, the assumption that unbroken encryption hides your activities is largely unchallenged outside of academic circles. This is certainly not the case given a motivated adversary and, as will be discussed, the effort required to perform such an analysis is perhaps rather low.

For purposes of compatibility, network protocols conceptually – although not always in practice – follow the OSI model [3]. This model denotes how protocols are separated into layers with specific responsibilities. Data of higher layer protocols is encapsulated within those below. The WiFi standard (IEEE 802.11 [4]) was adapted from the wired Ethernet standard (IEEE 802.3 [5]) and redefines the Physical layer (from a copper wire medium to 2.4 GHz radio broadcast) and Data Link layer (defining appropriate access control and collision detection for the new medium). Protocols in the layers above remain entirely unaffected so hardware and software does not need to be rewritten for every possible combination of protocols. For example, your web browser will fetch a web page using HTTP over TCP over IP regardless of whether a physical LAN or WiFi is used at lower layers. Internet-enabled smartphone apps are similarly agnostic as to whether they are connected via WiFi or the cellular data network for the same reason [6]. However, this results in a situation where protocols can operate in environments where the assumptions they were designed under may no longer apply. Data, activity patterns or other information can be exposed in ways the original implementers did not consider.

For the sake of confidentiality, WiFi networks should therefore employ an encryption scheme to prevent data being read directly. Even though the addition of encryption greatly improves security, WiFi still presents a large and accessible target to potential eavesdroppers. Communications can now be observed and recorded over a great distances where physical access and a literal wire-tap would have been required previously. Wired Equivalent Privacy (WEP), the first widely deployed WiFi encryption scheme, is therefore a misnomer. The security of physical and wireless network communications are not equivalent at all. Should an encryption scheme contain a flaw (as they have in the case of both WEP [7] and the original WPA [8]) then communications can be decoded easily, remotely and usually without trace.

In this paper we make no attempt to find or exploit flaws to break WiFi encryption directly. Instead we show what can be determined despite encryption working perfectly as designed. In an environment with correctly implemented encryption, users and network operators would be excused for assuming that privacy and confidentiality were assured unless the secret keys were discovered. We demonstrate that this is not the case.

2.2. *Mobile devices and mobile app privacy*

The abundance of private information stored on modern mobile devices is well-established [9]. Following the jump from 'dumb phones' to smartphones, mobile devices suffer from the same threats that previously targeted personal computers and are privy to the same personal data. As useful services are added to mobile platforms, malware that exploits these services and the data stored is soon to follow. This malware performs all the activities one would expect. For example; stealing personal data, sending spam, or ransomware. An interesting twist on the spam business model is to generate revenue by calling premium-rate numbers or sending SMS [10]. Similarly, the propensity to install apps on mobile devices rather than visit a website, plus easy-to-use virtual 'app stores' or 'markets', means installing software on mobile devices is easy and routine. This is in stark contrast to traditional computers where installing specialised software just to see a particular news source or check the weather seems very far-fetched. Given the wealth of personal information and functionality that could be available to unrestricted programmes, great efforts have been made to set up permissions systems to regulate app behaviour [10,11].

Modern Mobile devices are undeniably a huge repository of personal information. Seneviratne et al. explored this fact and found that they were able to identify personal traits such as gender, religion and ethnicity app identification with accuracy approaching 90% [12,13]. This paper attempts the same inferences, but without privileged access to the mobile device's files, or even the network it is operating on. We must attempt to identify the apps from fully encrypted WiFi traffic first.

The mobile aspect of these devices adds further privacy risks. The most efficient methods of routing network communications rely upon the easy and unique identification of the sender and recipient. These and similar unique identifiers pervade unprotected network communications [14]. Even from an external vantage point with WiFi encryption enabled, MAC addresses that uniquely identify devices are still freely broadcast. Commercial interest in wireless broadcasts has piqued in recent years with companies recognising them as a potentially huge data source for the multi-billion dollar Customer Relationship Management market. London's controversial "tracking bins" that included hardware to collect WiFi-enabled device identifiers as owners passed in the street to track their movements [15], and Westfield Groups' programme to perform similar shopping habits analysis with mobile phone identifiers [16]. Furthermore, mobile devices may also actively search for familiar APs and in doing so may broadcast the names (SSIDs) of recently used networks. These can be located using the same databases that aid mobile GPS navigation to determine the home, workplace and other locations important to the user of a mobile device [17]. Disposable Network IDs would work to inhibit analysis of this kind [18] and while it would make tracking specific users between sessions much harder, it would not prevent the analysis presented in this paper.

The privacy threat of external tracking and the security of personal data held on mobile devices is often considered separately. In our scenario these two concerns are combined. We demonstrate how it is possible to exploit WiFi side-channels to infer private user information without the need for a privileged position within the network or on the device. This renders safeguards like app permissions and encryption irrelevant. The distance over which this analysis can be performed may also be much greater than expected. Although most commodity WiFi devices have a range of up to 100 m, this is with omnidirectional antenna and includes the need to transmit. With increased signal strength and directional antenna, WiFi networks using have been operated effectively point-to-point over hundreds of kilometres [19]. While most WiFi hardware is of course incapable of this, the observer in our scenario only needs to receive transmissions.

2.3. Side-channels & private information inference

Side-channel analysis is famed for its use in breaking encryption but remains "an often-overlooked threat" [20]. Power analysis of Data Encryption Standard (DES) hardware [21] and timing analysis of the RSA algorithm [22] leaked enough information about the state of the encryption processes to undermine them. These side-channel leaks reliably allowed the secret keys used in the encryption process to be deduced in a very short time and therefore rendered the algorithms insecure.

However, as we demonstrate in this paper, side-channels do not have to be used to 'crack' encryption directly to reveal sensitive information. Observing the older implementations of the SSH login protocol showed that password length could be deduced from the size of the packets sent to the server and padding was added to secure the process [23]. The anonymity-preserving Tor network also added methods to obfuscate the timing and size characteristics of traffic after it was shown that visiting certain websites could be 'fingerprinted' [24,25]. Similarly, Chen et al. [26] investigated HTTPS side-channel leaks for secure cloud services handling personal healthcare and tax information. They found that "side-channel information leak is a realistic and serious threat to user privacy". Another study found that it was possible to deduce certain categories of Google search despite the user connecting over encrypted HTTPS to a distributed Content Delivery Network [27].

In mobile-specific work, Stöber et al. [28] looked to identify the apps signified by variations in their 'sync' (regular update) activity on Android devices. Using the same side-channels as this study, they were able to reliably identify which apps were being sync-ed on the device. However, the number of apps this method applies to is restricted to only the relatively few apps that use the sync mechanism. However, all these studies are undertaken from a position within the network. While the challenges in dealing with encrypted traffic are similar, any adversary must have a position of privileged access like that of a network administrator or Internet Service Provider. Observing passively from a perspective external to the network, Zhang et al. [29] found they were able to classify broad categories of encrypted WiFi traffic (*e.g.* video streaming, web browsing) and Atkinson et al. [30,31] showed it was possible to specifically target Skype traffic for identification.

Other studies examined inference one step further. In some cases it is possible to infer not just a private activity, but also what personal information can be inferred from that activity. For example, it was shown that the tiny but frequently updated information from 'Smart Grid' power meters could be used to not only infer the appliances used [32], but also how the usage of these appliances could then 'leak' further information about occupants' behaviour. For example; meal times, working hours and TV habits. Going even further, the authors hypothesise that certain behaviours can be correlated with predictable social timings – such as church or bar opening times – to expose even greater depths of private information [33].

Similar analysis can be performed on network communications. Impressively, White et al. [34] were able to not only accurately model encrypted Skype activity using side-channel properties, but also reconstruct entire spoken conversations from timing and size characteristics although it did not generalise well to variations in speech patterns (*e.g.* a different regional dialect). Saponas et al. [35] showed that WiFi-enabled streaming media devices could be observed to determine which film was being watched despite encryption and potentially leak the personal interests of the viewers. This paper adopts a similar strategy: by observing the activity of specific apps, we can infer personal information about their users. Furthermore, the mobile aspect of apps makes opportunities for observation by third parties are far more likely.

3. Methodology

3.1. Mobile app selection

We selected 34 apps that we would attempt to detect remotely despite WiFi encryption being employed. At the time of writing, the apps chosen could be found among the 'top' free app listings in their Play Store category. This is not Android-specific and most of these apps will have equivalents on alternative platforms. If we assume that users are representative of an app's majority demographic, Table 1 shows how the ability to detect the use of these apps allows personal information about their users to be inferred. Inferences may change depending on context so we assume these apps were identified in a public location such as a shopping centre in London, UK.

To assess the importance of the information being leaked, we loosely borrow from the categorisations in EU Data Protection legislation. Information classified as 'Personal' is that which could be used to identify an individual. While it is not possible to infer the canonical examples of this category (name and address), other personal information including gender, age range and nationality can easily be inferred for the typical app user. Information categorised as 'Sensitive' is personal information where the importance of additional confidentiality is recognised. If disclosed such information could be embarrassing, harmful to the well-being of the individual, or used as a basis for prejudice. Less personal, but still intrusive is where use of an app implies a 'Specific marketable interest or hobby'. While this may not be particularly important, it could easily be used to help sell products, target advertising, or otherwise uncover an individual's personal interests. Finally, some apps may require certain capabilities from a device in order to work (for example, geolocation or a HD screen).

It is important to recognise that these inferences are only generalisations based on an app's typical user demographic. They will likely hold in a majority of cases, but not all. The *accuracy* of these inferences can vary. The nationality of a given news source may correlate with its readers when generalised over all users. However, for a specific user this is not necessarily the case. An American may just like to read (UK-based) BBC News as a personal preference. Similarly, while the majority of AutoTrader users may be male, there are still plenty of female car enthusiasts and while most people using dating apps are single, some will be married. Conversely, the MySugr app is essentially useless to anyone who is not diabetic so it is highly likely that health information can be inferred about the user correctly. Furthermore, the *precision* of these inferences can also vary depending on the narrowness of an app's appeal. While it is impossible to infer the exact date of birth for a given user, an age range can be inferred depending on the app. For example, apps may target specific age ranges (*e.g.* college

Table 1
User information inferable from use of mobile apps.

		Pop.		News						Retail					Rel.		Lifestyle								Est.		Health				Ent.			Tra.	
	App Name	GMail	Twitter	BBC News	Daily Mail	Buzzfeed	Guardian	Le Monde	NY Times	Aldi	M&S	Game	Auto Trader	H&M	Bible	Al-Quran	Tinder	Grindr	Divorced D.	YPlan	Urban Spoon	Runkeeper	Sky Sports	Lottery	RightMove	Sotheby's	Expecting	Period T.	MySugr	Anxiety Utd.	Spotify	Shazam	Netflix	Trip Advisor	Nat. Rail
	No. Downloads	>1Bn	>100M	>5M	>1M	>1M	>1M	>1M	>5M	>100K	>100K	>100K	>1M	>5M	>50M	>1M	>10M	>1M	>10K	>100K	>1M	>10M	>1M	>1M	>1M	>10K	>1M	>10M	>10K	>5K	>10M	>100M	>50M	>50M	>1M
Personal	Gender											✗	✗	✗				✗					✗				✗	✗							
	Child / Adult			✗	✗		✗	✗	✗	✗	✗		✗		✗	✗	✗	✗	✗					✗	✗	✗	✗	✗						✗	
	Specific Age Range					✗											✗	✗	✗			✗			✗		✗	✗							
	Marital Status																✗	✗	✗																
	Income									✗	✗														✗	✗									
	Nationality			✗	✗		✗	✗	✗											✗															
Sensitive	Sexuality																	✗																	
	Religion														✗	✗																			
	Race / Ethnicity														✗	✗																			
	Physical Health																					✗					✗	✗	✗						
	Mental Health																													✗					
	Political Views				✗		✗																												
Specific Marketable Interest or Hobby	Travel							✗	✗																									✗	✗
	Dining															✗					✗														
	Grocery									✗	✗																								
	Entertainment																			✗											✗	✗	✗		
	Fashion										✗			✗																					
	Sport / Fitness																					✗	✗												
	Gambling																							✗											
	Technology											✗																							
	Automotive												✗																						
	Dating																✗	✗	✗																
	Property																								✗	✗									
Other	Device Capability																	✗				✗											✗	✗	

Notes: *(1) YPlan more accurately implies place of residence, however for easier categorisation we denote this as 'nationality'.*
(2) Inferences are a generalisation of user demographic with varying accuracy and precision. Refer to text for more detail.

students), and many apps provide information that is only of use to adults (*e.g.* real estate) and there are many apps designed for children (although we do not include any).

Popular apps like GMail and Twitter are so widespread that their use signifies little about the user. News apps on the other hand can signify nationality based on their origin (BBC News is British, and New York Times from the US) or language (Le Monde is French and written in French), and the likely political leanings of the reader (The UK newspapers Daily Mail and Guardian are considered right- and left-wing respectively). BuzzFeed is social-media based and attracts a certain younger demographic. Retail apps can inform a great deal depending on their market specialisation. We chose several from which a user's likely income (Aldi and Marks & Spencer are UK supermarkets at low and premium end of the market respectively) or gender can be inferred (The audience of AutoTrader, a car sales app, and Game, a computer games retailer, will be predominantly male. H&M's app focuses on women's clothing). Religious apps imply the likely religion of a user that can be linked to common racial or ethnic trends.

Lifestyle apps cover a wide range of personal interests. Dating apps inform on the probable marital status of the user and age range ('young and single' in the case of Tinder or Grindr). Grindr further specialises to target gay men, and Divorced Dating provides a dating service to a even more specific marital status. YPlan is an events recommendation service limited to 5 major US and UK cities at the time of writing. It implies a likely nationality, or more accurately a likely area of residence. Urbanspoon provides information on nearby restaurants, Runkeeper is a map-based fitness tracker and Sky Sports provides sports news. The Lottery app is used to check numbers from the UK's (age-restricted) national lottery, and implies a positive disposition to gambling. Real Estate apps provide information on the income of a user (*i.e.* wealthy and old enough to wish to buy a house) with Sotheby's focusing on the very wealthy especially.

Health apps can relay private health information with MySugr for diabetics and Anxiety United's app for those who suffer from anxiety. Period Tracker (a menstruation diary) and I'm Expecting (a pregnancy app) not only denote health information but also specific gender and age ranges. Like Twitter and GMail, as a common activity Entertainment and Travel apps belay little information with the possible exception that travellers are likely to be (or accompanied by) adults. However, they do imply the marketable information that the user has free time at a location and device capabilities such as HD screen and fast internet connection.

3.2. *Measuring app activity*

Here we demonstrate the ability to build a classifier that identifies app usage from the perspective of Fig. 1's external observer. The classifier will operate in a scenario without many of the common network data that act as service identifiers. The likes of DNS queries, IP addresses, port numbers, and more are hidden behind encryption from the observer's perspective. However, the classifier itself need not be built from data collected from this vantage point. Previous classification efforts attempted to sample network data directly from this position [29,31,35]. Although identical to the scenario that the classifier would operate, this makes recording accurately labelled samples difficult to coordinate because the collection platform and device creating the network traffic have no direct communication method. Synchronising the capture process with the desired activity depends on the accuracy of careful timing or artificially-inserted indicator packets.

To avoid the complications of synchronising an external capture mechanism with on-device app activity, a method was developed to capture network traffic locally (on the device) then transform the information to how it would appear from an external perspective as the observer would see it. The 'WiFinspect' [36] app was used for this purpose. As an Android-specific interface for the commonly known TCPDump [37], it saves network data to standard PCap (packet capture) format for later processing. The same local sender and receiver MAC addresses, frame size and timing data that would also be available from an external

Please cite this article in press as: J.S. Atkinson, et al., Your WiFi is leaking: What do your mobile apps gossip about you?, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.05.030

perspective is extracted using TShark from the well-known Wireshark packet analysis suite [38]. The remainder of the data can be discarded.

However, we found that this data more than sufficient for our purposes and greatly simplified the data collection process so that clean data could be easily gathered in large quantities. We used Android devices during practical experimentation due to greater tool familiarity, however it is equally possible to collect data in similar ways on other mobile platforms (*i.e.* iOS or Windows Phone) provided sufficient privileges required to run packet capture software could be obtained. This usually requires device rooting.

The network activity observed will differ slightly from the activity that would be recorded naturally. For example, information on Data-Link layer frame retransmissions will be lost and timing data will vary due to processing delays. A particular device's network hardware and OS network stack will also affect exact timings. The exact distribution of these delays vary by device [39]. However, they should be no more problematic than the already variable Round Trip Times (RTTs) inherent in internet communications [40] and variation over the 3 different network configurations we use. The use of cumulative metrics helps mitigate this issue as described below.

We recorded the network activity when launching each app. This will load some sort of main page or welcome menu. Depending on the app, this might trigger network activity for a login or authentication process, fetch initial content (*e.g.* latest news or location-sensitive content), display ads, or report usage statistics to the developer. Internally, the recorded network activity is predominantly HTTP or HTTPS (*e.g.* for authentication) with DNS alongside to resolve hostnames to useable IP addresses. These specifics are not used by our analysis as they would be hidden from an external observer by encryption. However, they are useful to understand the typical activities of an app at startup. In the apps surveyed we found that even those with entirely static local content (like Al-Quran, and Shazam's home screen) used network communication at startup when connectivity was available.

Recorded over several weeks with automatic software updates disabled, our mobile devices were controlled via USB connection in conjunction with automation software [41]. We therefore simulated the user actions required to begin packet capture, open an app, wait until loading is complete, then halt the packet capture process. This process can then be replayed to generate samples of app network activity without direct human interaction. Opening the app is a common, essential but very simple action. This simplicity is designed to minimise any variation arising from the differences between automated and human action. This on-device data is sufficiently close to the off-device equivalent and TShark's was configured so that external frame payloads are consistently reported as 16Bytes larger than their on-device equivalent. By simply adding an offset to the 'data length', these measurements provide representative characterisation of app activity from an external observer's perspective. These measurements can then be analysed to differentiate mobile app network activity.

In an attempt to build a classifier that would not overfit to the characteristics of a particular network, samples were collected from three different types of network. The first was a home network employing a standard ISP-supplied router to provide internet access over ADSL. Other devices on the network included several mobile phones, laptops and media streaming devices. The second network was the University enterprise-grade network. This utilised multiple Cisco APs and would operate with over ten connected devices (predominantly phones and laptops) at any time. The third was a WiFi network supplied from a 4G 'MiFi' dongle providing internet access via a LTE cellular network. Aside from the tablet whose traffic was being recorded, this network had no other connected devices. All networks used WPA2 (with PSK authentication for home and 4G networks, and PEAP-MSCHAPv2 for the enterprise network). Our devices connected at 802.11g speeds, although all APs also provided b and n data rates.

In total, 7480 app samples were recorded. The 220 recordings per app consisted of 120 recordings from the home network, plus an additional 50 each from both 4G network and enterprise networks. Additionally, 1766 samples of other activities were used. These recordings came from previous work and were composed of a wide variety of activities including Skype, BitTorrent, web browsing and idle time, but importantly no mobile apps. This allowed for the production of a classifier able to predict "activity other than that of a known app" (denoted 'OTHER'), and not just be forced to classify activity as one of the 34 apps. Without this ability, false positives on any real implementation would be entirely unavoidable. The generalisation error of this OTHER class label would also provide a lower limit for false positive detection rates.

3.3. Random forest construction

To construct a classifier from the available data, we employed Random Forest machine learning using the BigRF package in R [42,43]. Popularised by Breiman and Cutler [44], Random Forests are a supervised, non-parametric machine learning technique and therefore rely on carefully labelled sample data. As will be detailed shortly, our samples from encrypted network activity are labelled as belonging to a particular app (or OTHER). After construction the Random Forest will be able to predict the app in use (or OTHER) when given a sample of encrypted network activity.

The accuracy of this prediction process is measured as Out-Of-Bag (OOB) error. This is a generalisation error based upon the ability for the Random Forest classifier to correctly classify sets of test samples. Unlike many other machine learning algorithms, Random Forests do not require manual separation of data into these Training and Test sets. This is performed as part of the construction process. Each tree selects approximately 2/3 of samples for construction at random and leaves aside 1/3. Unlike machine learning methods that evaluate error with a single test following classifier construction, it is therefore considered unbiased [44].

Random Forests are an ensemble method, formed by building many Decision Trees. Comparable to the common flowchart, a Decision Tree is a sequence of nodes and directed paths ('branches') used to predict the class of a given sample. Any route through the tree starts at a single 'root' node and eventually ends at a terminal 'leaf' node providing an app prediction. The route (and therefore the final prediction) taken depends on the outcome of decisions at each node. Each node compares a variable from the sample data with a value (as will be demonstrated in the next section). The outcome of this comparison then determines the branch taken to the next decision node or terminal leaf.

The best decisions to have at each node are determined as part of the Random Forest construction process. The measurements that form each labelled sample allow differences between classes to be characterised. Each decision attempts to separate the apps based on their network activity. Providing the tree is well constructed, the sequences of decisions will cause the available classes to gradually separate as the path through the tree is followed. A Decision Tree within the Random Forest is grown as follows [45]:

1. Generate a new random training subset for each tree: *N* training samples with known class labels are taken from the complete dataset with replacement (a sample can be selected multiple times).

2. Grow the tree: At each node in the tree, select a different random selection of m variables from the M total in each sample. The best decision on m to split the N samples by classes is calculated and used for this node. The value of m is constant across the entire forest.
3. Terminate with leaves denoting a class: Each tree is grown to the largest extent possible. Decision nodes are grown until only a single class remains on a branch. This branch will then lead to a terminal 'leaf' node denoting the relevant class label.

The Decision Trees that constitute the forest attempt to optimally separate the classes based on random subsets of the variables and samples provided. Individually their ability to generalise is poor due to working with only a small fraction of the available data. However, their predictive power is much improved when used in combination. This ability is the foundation of ensemble machine learning methods, where combining many smaller less accurate classifiers can be easier to construct and just as powerful as a single monolithic classifier.

3.4. Available data and metric distributions

This method to infer personal information makes no attempt to break WiFi encryption directly. This results in a highly data-limited scenario. WiFi encryption is applied at the Data-Link layer. Therefore the only data available is that contained in the frame header between device and AP, plus any side-channels we can measure. We can therefore only gather the following:

- Direction (from broadcast Sender and Receiver MAC addresses in the 802.11 header)
- Frame Size (measured or read from header)
- Interarrival Time between frames (measured).

However, these measurements can be represented as value distributions (histograms) and these distributions can be used to characterise different network activities. For each of the 220 samples of each app, the distributions listed in Table 2 were created. These distributions come in 3 broad categories: Frame Size, Interarrival Time, and Cumulative Interarrival Time. Information from these Frame Size and Timing measurements can then be further refined by separating this data based on the direction of the observed frames.

The Frame Size (FSize) distribution is created by simply plotting a histogram of the size of each frame (in bytes). To distinguish sent and received data, outgoing and incoming frame sizes are plotted separately. Unlike timing measurements, Frame Sizes are discrete, exact and not expected to exhibit natural variation due to jitter, delay or measurement error. Frame Size can be therefore plotted directly. For efficiency, frame sizes that occur in fewer than two of the app traffic samples are not provided to the Random Forest construction process. This is a very low threshold but nevertheless results in the removal of 1517 redundant variables so that only approximately half of the 3200 FSize variables are provided to the Random Forest. This reduces the time required to build the Random Forest (otherwise the algorithm would have to learn the unimportance of these variables itself). The size of this reduction is entirely data dependent. For other datasets (*e.g.* different fingerprinting scenarios, or additional apps) the ability to perform this reduction may be much harder or easier.

Interarrival distributions plot the time period between frame observations. As a continuous variable, time must be 'binned' into ranges to form a histogram. We used bins of 3 ms, from 0 to 150 ms. Again, this information can be partitioned using the direction of each frame. The combinations of interarrival time measurements between frame direction result in four distributions: Time between Received frames and the previous Received frame (I-RR); Time between Received frames and the previous Sent frame (I-SR); Time between Sent frames and the previous Received frame (I-RS); Time between Sent frames and the previous Sent frame (I-SS).

Table 2
Distributions created for each 15s window.

Distribution	No. variables	Description
FSize	3200 → 1683[a]	Frame Size (1600B per direction)
I-RR	50	Rcvd-to-Rcvd interarrival timing
I-RRCum	50	Cumulative transformation of I-RR
I-SR	50	Sent-to-Rcvd interarrival timings
I-SRCum	50	Cumulative transformation of I-SR
I-RS	50	Rcvd-to-Sent interarrival timings
I-RSCum	50	Cumulative transformation of I-RS
I-SS	50	Sent-to-Sent interarrival timings
I-SSCum	50	Cumulative transformation of I-SS
Total	**4000→2483**[a]	

[a] FSize subset selected prior to RF construction. See Section 3.4.

Cumulative Interarrival Time plots for each of the Interarrival Time distribution combinations are also generated as an alternative representation of the interarrival data. They are identical in construction, but include the sum of previous (lower interarrival measurement) bins in addition to the value of the bin itself.

The combined output of these distributions is an array of 2483 integer variables after processing. These distributions can be generated from labelled recordings of network activity and used to construct the Random Forest classifier. Alternatively, they can be generated from observed network traffic (recorded or live) and provided to the Random Forest to predict the app activity within. Differences in these distributions per app can be used to differentiate between the app in use for a given sample. To demonstrate this, Fig. 2 shows the most important variables within each distribution type (Frame Size, Interarrival, or Cumulative Interarrival) and plots their respective values for every recording of 6 different apps. The ability to rank variables by importance is a useful feature of the Random Forest algorithm.

For example, in most samples the number of sent frames observed with a size of exactly 66 bytes can be used to separate the BBC and Spotify activity from the other apps shown. The Random Forest could use the decision $FSize_{66} > 0$. The number of 0–3 ms interarrival measurements between received frames and the previous sent frame could then further split the majority of BBC activity from Spotify activity with a decision like $ISR_1 > 70$. Similarly, the cumulative number of interarrival measurements between sent frames that are less than 150 ms, $ISSCum_{50} > 250$, can separate the majority of Tinder samples from Bible samples where the other variables cannot.

Conversely the patterns shown by The Guardian, RunKeeper and Tinder are much harder to separate using only these variables. The decisions to classify these apps can be much more convoluted and this is why the Random Forest algorithm is used. Of course, the Random Forest has the entire set of variables to work with but must classify and separate the characteristics of 34 apps plus other traffic, not just the 5 apps and 3 variables chosen as examples.

4. Classifier results

Table 3 shows the generalisation error for each app and OTHER traffic over the collected dataset. As mentioned earlier, it is important to note that Random Forests do not manually define Test and Training sets. Therefore, OOB error is equivalent to the generalisation error reported on the Test set in other machine learning methods.

Using our method we found that 14 of the apps were identifiable with 100% accuracy. The remainder all had less than 9% error on this sample data with an average app identification rate of ∼98.5%. The hardest app to identify was GMail with 7.3% error. GMail was also the app with smallest mean sample filesize.

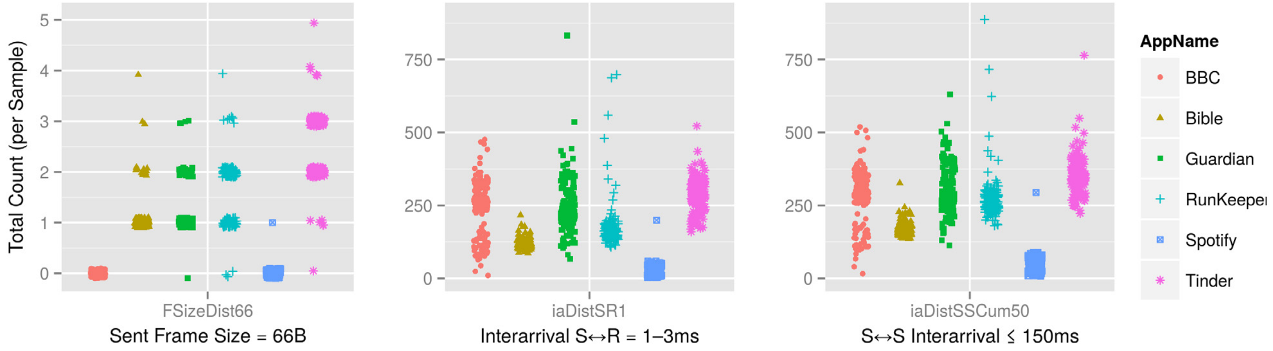

Fig. 2. How apps may be differentiated via distribution variables.

Table 3
Random forest generalisation error.

	Pop.		News						Retail					Rel.	
App Name	GMail	Twitter	BBC News	Daily Mail	Buzzfeed	Guardian	Le Monde	NY Times	Aldi	M&S	Game	Auto Trader	H&M	Bible	Al-Quran
Mean Sample Size (kB)	11	172	322	1101	138	307	1944	163	134	670	110	54	255	117	199
OOB Error (%)	7.3	0.6	0.6	0.0	3.0	0.0	0.0	0.6	1.2	0.0	0.6	0.0	0.6	0.0	3.1

	Lifestyle								Est.		Health			
App Name	Tinder	Grindr	Divorced D.	YPlan	Urban Spoon	Runkeeper	Sky Sports	Lottery	RightMove	Sotheby's	Expecting	Period T.	MySugr	Anxiety Utd.
Mean Sample Size (kB)	372	514	11	179	48	192	76	108	17	104	79	58	84	80
OOB Error (%)	0.6	0.0	3.1	0.0	0.6	0.0	0.0	0.0	3.1	0.0	3.8	1.9	0.6	2.5

	Ent.			Tra.		
App Name	Spotify	Shazam	Netflix	Trip Advisor	Nat. Rail	Other
Mean Sample Size (kB)	35	87	107	57	34	—
OOB Error (%)	0.6	0.0	0.0	2.6	1.9	8.8

App Detection Accuracy – Min: 91.23%, Max: 100%, Mean: 98.54%. False Positive Rate – 8.8%

Plotting this data across all sampled apps shows a general trend of lower error rates as average sample file size increases. This is perhaps unsurprising; with more bytes of network data to represent an app's startup activity, the more characteristics can be discovered and compared to distinguish that activity from others.

While correct app identification rates are very high, the usefulness of the classifier is limited unless it can be combined with a low false positive rate. The generalisation error for our OTHER traffic was 8.8%. In real world scenarios the maximum accuracy possible is therefore greater than 90%. This is encouragingly high and leaves room for error in real-world scenarios which will inevitably present more noise and varied activity than our sample data.

We have shown that the use of specific apps can be accurately 'fingerprinted' and identified even when encryption is being used. As illustrated earlier in Table 1, we can therefore also infer personal information about the users of these apps when they are detected.

5. Personas and real-time activity detection

Having created an accurate classifier, this section presents a study into how well the classifier works in real-time on more varied and noisy real world WiFi communications. A live analysis was then performed in which we attempted to identify the apps corresponding to 7 different personas. Personas are a common user-experience testing and marketing technique that allow the creation of a realistic (but fictional) representation of a human individual. As with real individuals, the personas have their own interests and potentially sensitive characteristics. These characteristics are expressed in their use of mobile apps as detailed in Table 4. The detection of these apps by a passive observer (and therefore personal information inference) is demonstrated in real-time and despite WiFi encryption. The Random Forest classifier created in R was ported, compiled in C++, and directly coupled to TShark to increase performance. The live detection application is fast enough that it can analyse the activity of multiple devices in real-time simultaneously. However, with a single WiFi adapter this is limited to devices operating on the same channel.

Table 4
Personas and app-signified characteristics.

Persona name	Characteristics (*Signifying app*)
Prof. Plum (6 apps)	A very wealthy (*M&S*, *Sotheby's*) British (*BBC*) Muslim (*Al-Quran*). Commuting (*Nat. Rail*) academic with diabetes (*MySugr*).
Miss. Scarlett (4 apps)	Single (*Tinder*) female (*H&M*) young adult. Loves music (*Spotify*, *Shazam*).
Col. Mustard (4 apps)	Conservative (*Daily Mail*) member of the British (*Daily Mail*) armed forces. Likes fitness (*RunKeeper*), football (*Sky Sports*) and bets occasionally (*Lottery*).
Mrs. Peacock (4 apps)	French (*Le Monde*) tourist (*Urban Spoon*, *Trip Advisor*) travelling via train (*Nat. Rail*).
Mrs. White (3 apps)	Divorced (*Divorced Dating*) commuter (*Nat. Rail*) and woman of child-bearing age who is pregnant (*I'm Expecting*).
Rev. Green (3 apps)	Christian (*Bible*) man looking to buy/sell his car (*AutoTrader*). Resident of London or nearby area (*YPlan*).
Dr. Black (3 apps)	Gay male (*Grindr*) of above average income (*M&S*) suffering from anxiety (*Anxiety Utd.*).

As with training data collection, the activities of personas could be automated for easier repetition. During testing personas would take a total of 10 actions, actions included opening the apps belonging to that persona exactly once. In addition to the apps from our 'closed-world', the remaining 'Extra' actions would other activities that used the WiFi connection. 'Extra' activities would be to visit one of a selection of websites at random using the mobile chrome browser or use another app not included in this study and provide an 'open-world' scenario. Personas would perform all ten actions in a random (but reported) order.

Every second, the live detection program would take all network activity observed in the previous 15 s and attempt to predict the app activity within the traffic for each monitored device. From our original activity observations, 15 s was the upper limit of the time required to automatically start recording, open an app, then stop recording. Without the overhead of starting and stopping recording and no automation software delay, a real user would be able to complete these actions much faster. Given that

Table 5
Live persona detection results.

	Monitored activity	Activity count										
		Apps	Idle	Extra	TP	FP	FN	TN	FPR (%)	Precision (%)	Recall (%)	Accuracy (%)
Closed-world	Apps & Idle	540	1400	–	466	229	74	1171	16.4	67.1	86.3	84.4
Open-world	Apps & Extra	540	–	860	466	619	74	241	72.0	42.9	86.3	50.5
	All	540	1400	860	466	848	74	1412	37.5	35.5	86.3	67.1

detection process was performed every second, it was determined that the typical app startup activity would usually register 8 s of positive detection. As a conservative estimate we therefore used a 5 s minimum threshold for a positive identification of an app. This helps eliminate spurious momentary app identifications. These are common in the few seconds before an accurate true-positive identification. At these points the Forest is recognising that something besides OTHER traffic is present. However, with only part of the information because opening the app is still in progress, the consensus as to which app is present is incorrect. As we know that opening an app will cause detectable network activity over a period of many seconds, these spurious identifications of 4 s or less can be discarded.

The 10 actions of each persona were repeated 20 times. With 3–6 apps each these provided a total of 540 time periods where mobile apps were launched. The remainder of the observed activities were 860 additional activities ('Extra') that used the network. Following each activity was a period of idle time before the next activity began. This corresponded to an additional 1400 'idle' time periods with no network activity other than that caused by background processes. 'Apps' and 'Idle' activities were all contained within the training data and therefore constitute a closed-world scenario. The addition of 'Extra' activity provides an open-world test.

We consider these time periods as single units to provide an empirical measure of performance as shown in Table 5. The detector's output for each time period was recorded. A true positive (TP) denotes a period of app activity where the app was correctly identified. Identification of any app in 'idle' or 'extra' time periods would be a false positive (FP), otherwise this would be a true negative (TN) with no app detected (for longer than the detection threshold). False negatives (FN) correspond to a period of app activity occurring but not being detected. Measurement in this way groups all app classes together. Although more concise, this does not show the case where app activity is detected but the wrong app is identified. In Table 5 these are also counted as false negatives and discussed further below. Similarly, it is important to note that true positives not only denote correct detection of an app, but also which one.

The initial closed-world scenario allowed us to demonstrate detection succeeding in spite of data limitations. We were able to identify apps despite their communications being entirely encrypted. In the closed-world the classifier was only required to detect activities it had previously encountered. If the activity of a targeted app was present then it was correctly identified in 86.3% of cases despite encryption (high recall). Of the 74 false negatives 30 (41%) were app activity identified as the wrong app (with the remainder being identified as OTHER). A false positive rate of 16.4% results in an overall closed-world accuracy of 84.4%. This entails a gross lack of privacy for our personas. For example, traffic patterns from Dr. Black's device would imply that he is most likely relatively wealthy, gay, male, and an anxiety sufferer provided he opened the M&S, Grindr and Anxiety Utd apps whilst under observation. Although there are occasional false positives, multiple observations of the same activity on the device would increase the certainty over time.

When considering the open-world scenario where 'Extra' activity (previously unseen by the classifier) was also included, a substantial increase in false positives was observed. For 'Apps' and 'Extra' activity alone (*i.e.* excluding FPs from 'Idle' activity) 'Extra' activities produced a greatly increased false positive rate of 72%. Overall accuracy for all activity drops to 67.1% in the open-world case and only 50.5% when considering only 'App' and 'Extra' activity. Precision similarly suffers due to this increased false positive rate at 35.5% and 42.9% respectively. In this open-world scenario Miss Scarlett's device would again identify her as a potential single female who enjoys music via use of the Tinder, H&M, Spotify and Shazam apps. However, the increased number of false positives for all apps gives the inference less credibility. A far greater number of observations of the Tinder, H&M, Spotify and Shazam apps would need to be made to provide the same level of confidence in the personal information inferences compared to the closed-world.

The small number of false negatives means that a negative detection remains a strong prediction that an app was not used in either scenario. Unfortunately, this is not very useful information in context. For example, although Prof. Plum's use of the Al-Quran app strongly implies an interest in the religion of Islam, the converse is not true. Prof. Plum may still be Muslim and not use the app. However, low false negatives may be useful in other situations such as the forensic scenario discussed in the next section.

6. Discussion

Detection in terms of positively identifying the targeted apps was largely successful. If an app's network activity was truly present in live traffic, then the app would be identified with high recall (true positive rate) and any personal information associated with that app was demonstrably leaked to the surrounding area. This means that identifiable patterns that betray personal information are broadcast from mobile devices whenever an app is used despite the use of encryption. Sensitive information including age, religion, sexuality and gender is there to be collected for anyone listening.

However, in an open-world scenario any positive result from the detector becomes less meaningful. As indicated by the low precision score, without improvements to decrease the false positive rate any positive result has low predictive value. The usefulness of these results as they stand depends on the scenario and application.

6.1. Prevention and applications

Although we have demonstrated the ability to infer private user information despite WiFi encryption, existing techniques to thwart web fingerprinting methods will still work to mask the app detection shown in this paper. Specially designed VPNs [46] and anonymity networks such as Tor [47] pad frame sizes, adjust timings, and intermix network traffic. However, these methods do incur significant performance penalties. A network with optimal throughput will attempt to send data as fast as possible only when required (causing predictable timings), and only as much data as is needed (causing predictable frame sizes). Shifting priority away from maximum throughput introduces significant overhead by necessity, and is especially problematic for mobile devices where performance is at a premium due to battery life limitations.

Please cite this article in press as: J.S. Atkinson, et al., Your WiFi is leaking: What do your mobile apps gossip about you?, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.05.030

While these solutions exist, they are unlikely to see widespread deployment on consumer devices in the short or medium term.

As noted in Section 2.2, mobile devices are already used to track customer habits. Provided improved false positive rates can be achieved, it would be possible to infer personal demographic information from local WiFi users. This kind of inference underpins the modern practice of "targeted advertising" [13,48]. However, while shopper footfall tracking may be tolerated, the legality and ethics of monitoring device communications to add personal data to this process is questionable and will vary by jurisdiction. Although not ideal, the simplest solution for concerned users is to not use apps with an association to sensitive information in public areas where they are likely to be monitored. However, even if 100% accurate app detection methods could be achieved, an observer may still have to procure exact user demographic data for an app and contend with the problems of inference accuracy and precision discussed in Section 3.1.

Law enforcement may also have uses for similar activity inference, with encryption becoming more common by default. This is especially true if analysis can be performed in real-time as it may be able to indicate when a particular system or application is active leaving volatile data is accessible and encrypted data unlocked. Similarly, this approach may be a useful supplement to the forensic analysis of encrypted network traffic in bulk after-the-fact. The ability to detect specific activity without breaking encryption can help focus a forensic investigation on certain devices and time periods without needing to break encryption. Even with false positives, this may be a more attractive and tractable prospect than attempting to break the encryption of vast quantities of streamed data in bulk before it can be analysed.

Finally, this research may lead to increased awareness of how the privacy guarantees of WiFi encryption are perhaps weaker than expected. We hope this work will spur interest and the development of countermeasures appropriate for mobile devices, such as Tor's Orbot [49]. Awareness of this security weakness can allow privacy-conscious users and organisations to recognise where inference techniques such as this may be a security or privacy risk and change their usage if appropriate.

6.2. Detector improvement

The greatest impediment to a low false positive rate were 'Extra' activities. While the data from our OTHER dataset contained a variety of common web traffic and allowed the Random Forest to usually discount idle (and near-idle) traffic and some of the added activities correctly, it proved an insufficient baseline to accurately discount the universe of all traffic outside our targeted apps' activities. Future work should look to utilise a larger and more representative dataset of other traffic. It may also be sensible to consider a two-tier classification process whereby interesting activity (*i.e.* apps) is first separated from all other activity using a simpler Boolean classifier before any samples classified as 'interesting' are passed onto a second classifier specialising in identifying the correct app. This would partition the problem into two separate questions that could be tackled separately.

The data used in this study was only sourced from a single WiFi data rate at 802.11g (as opposed to 802.11a, b or n). Changing the rate at which packets could be sent would not affect frame sizes, but would undoubtedly affect interarrival timings. While the classifier is unlikely to operate well one different speed WiFi networks as currently calibrated, we did demonstrate that data for 3 different types of network allowed for generalisation of detection over all three. Given that classification operated over a variety of network types (particularly the 4G dongle), we can have confidence that the techniques presented in this paper would be just as effective if training data from networks with different data rates was sourced.

Furthermore, assuming that frame size and interarrival metrics are common to both, we can also have confidence that the same mechanism could be adapted to 4G LTE. This widens the privacy risk to another widely used and longer range mobile protocol, but would require more complex hardware for traffic observation (unlike WiFi, the upload and download protocols are asymmetric as noted in Stöber et al.'s work). Similarly, although we can monitor any number of devices, our observations were limited to only observing a single WiFi channel (frequency) at a given time. This is a hardware limitation, but could be overcome by using multiple WiFi adapters in parallel.

6.3. Practical threat development

Data collection is the main challenge to the development of an inference system like this. The predictions can only ever be as good as the sample data provided to the classifier construction process. Aside from the previously discussed larger dataset of non-interesting traffic to reduce false positives, every app (or other interesting activity) must have sufficient sample data for every permutation of the identification problem. This will require additional data for every change to the scenario. Random Forests will not be able to accurately classify something that was not provided as part of the training process: this approach remains 'signature-based'.

As noted earlier different data rates will affect interarrival timings. Although we demonstrated that signatures could be automated and generalised by sampling a variety of networks with different throughput capabilities, these samples still need to be collected. Targeting different WiFi data rates or cellular data protocols like LTE will require additional data collection for each network. Alterations to app programming are a potential problem for the longevity of an app classifier. Typically on-device apps are automatically updated when a developer publishes to the online app store or marketplace. While we demonstrated that changes in content could be overcome by exploiting other similarities (*e.g.* changes to the stories in News apps, or different active users on dating apps), automatic updates were disabled for the purpose of this study. The impact of an update will be dependent on its effect (or lack thereof) on the app's network communications.

For example, while stories featured on the BBC News app change routinely the ability to detect the app was retained. However, following the experimentation the Al-Quran app was allowed to update. This update changed the ad library in use and caused such major changes to the app's network activity that it was no longer accurately identifiable. Data collection must occur in real time and is therefore the most time-consuming part of the classifier construction process. Ruiz et al. [50] estimate that "51%–60% of free Android apps have at least one ad or analytics library", and that updates to the frameworks the sampled occurred every 1–3 months. Within our dataset only GMail, Twitter, BBC, MySugr and Aldi did not use a third party library. The remainder all used at least one with some appearing many times (*e.g.* Scorecard Research (7 apps), Crashlytics (7), Doubleclick (7), Flurry (6), Crittercism (3), Facebook (3)). This widespread use is encouraging in that it has not prevented detection despite reuse of libraries over different apps. Future work could investigate the variance in signature longevity in different apps as well as to what degree shared libraries complicate the process. Despite this, the collection of data to create these signatures can be largely automated and parallelised. User activity simulation and the ability to convert on-device recordings is far more practicable than collecting data from an external vantage point. We hope that future work can investigate exactly how much variance exists between developers utilising the same 3rd-party libraries and exactly how they are integrated into the software. Unfortunately most apps are closed source making this significantly harder.

Although popular, the 34 apps detailed in this paper only represent a small fraction of entire mobile app market. While free-to-download apps like the ones used in this study form approximately 90% the mobile app market [50], there are many paid apps. For these apps, serving ads or selling information about their users and is a not their primary business strategy. They may therefore be less reliant on internet communication. As was seen with Gmail's small network activity being more difficult to reliably fingerprint, less or no network activity may make paid apps harder to identify. Furthermore, there will be many common libraries and code shared across apps on the market as a whole. If this common programming causes network communication – as would be the case with common functionality such as serving ads, developer feedback, and mapping services – the resultant activity may be difficult to differentiate. Using related activities as contextual clues may help mitigate the problem of different app actions producing similar signatures. For example, in the Tinder dating app composing a message must be preceded by first opening the app and then viewing a profile. Tools such as NetworkProfiler [51] could allow for all user actions (and therefore all network activity) paths to be mapped through the app programmatically.

Finally, this study only considered scenarios where one activity was present at a time. This was seen an appropriate assumption given the modal nature of mobile apps and is a common assumption in related fingerprinting studies [25]. Although previous work has shown it is possible [31], simultaneous activities present a much harder problem. Simultaneous downloads and other background processes will affect network activity, altering the observed characteristics and adversely affect detection accuracy.

7. Conclusion

This paper demonstrates that private information including the likes of age, religion, sexuality and gender can be inadvertently broadcast by mobile device apps despite WiFi encryption being used as designed. We presented a remote, undetectable, detection mechanism to infer private user information through observation of encrypted app network activity. This has been demonstrated to work in real time, and with appropriate hardware should generalise to other encrypted communication methods. With longer range wireless communications becoming more prevalent, and commercial enterprise becoming more interested in tracking and analysing publicly broadcast wireless data, this paper highlights a plausible and demonstrable threat to users' privacy.

Acknowledgements

The authors would like to thank UCL SECReT (EPSRC Grant EP/G037264/1) and Selex ES for their involvement and support of this project.

References

[1] Google Inc., On vehicle-based collection of wifi data for use in Google location based services, Apr. 2010. http://static.googleusercontent.com/external_content/untrusted_dlcp/www.google.com/en//googleblogs/pdfs/google_submission_dpas_wifi_collection.pdf.

[2] Apple Inc., iOS 5: Understanding Location Services, Dec. 2010. http://support.apple.com/kb/HT4084.

[3] H. Zimmermann, OSI reference model-the ISO model of architecture for open systems interconnection, IEEE Trans. Commun. 28 (4) (1980) 425–432.

[4] IEEE-SA, Wireless LAN Medium Access Control, MAC, and Physical Layer, PHY, Specification, IEEE Standards Authority, 2007.

[5] IEEE-SA, IEEE Standard for Ethernet, IEEE Standards Authority, 2012.

[6] A. Ghosh, R. Ratasuk, B. Mondal, N. Mangalvedhe, T. Thomas, LTE-advanced: next-generation wireless broadband technology, IEEE Wirel. Commun. 17 (3) (2010) 10–22.

[7] A. Bittau, M. Handley, J. Lackey, The final nail in WEP's coffin, in: IEEE Symposium on Security and Privacy, (S&P'06), IEEE, 2006, pp. 386–400.

[8] E. Tews, M. Beck, Practical attacks against WEP and WPA, in: Proceedings of the Second ACM Conference on Wireless Network Security—WiSec'09, 2009, p. 79.

[9] K. Shilton, Four Billion Little Brothers? ACM Queue 7 (2009) 40.

[10] A.P. Felt, M. Finifter, E. Chin, S. Hanna, D. Wagner, A survey of mobile malware in the wild, in: Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices—SPSM'11, 2011, p. 3.

[11] M. Kuehnhausen, V.S. Frost, Trusting smartphone apps? To install or not to install, that is the question, in: 2013 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support, CogSIMA, 2013, pp. 30–37.

[12] S. Seneviratne, A. Seneviratne, P. Mohapatra, A. Mahanti, Predicting user traits from a snapshot of apps installed on a smartphone, SIGMOBILE Mob. Comput. Commun. Rev. 18 (2) (2014) 1–8.

[13] S. Seneviratne, A. Seneviratne, P. Mohapatra, A. Mahanti, Your installed apps reveal your gender and more!, in: Proceedings of the ACM MobiCom Workshop on Security and Privacy in Mobile Environments, SPME'14, ACM, New York, NY, USA, 2014, pp. 1–6.

[14] T. Aura, J. Lindqvist, M. Roe, M. Anish, Chattering laptops, in: Privacy Enhancing Technologies Symposium 5134, July, 2008, pp. 167–186.

[15] J. Vincent, London's bins are tracking your smartphone, in: The Independent, London, UK, 2013.

[16] A. Censk, Malls track shoppers' cell phones on Black Friday, Nov. 2011. http://money.cnn.com/2011/11/22/technology/malls_track_cell_phones_black_friday/index.htm.

[17] G. Wilkinson, Digital Terrestrial Tracking: The Future of Surveillance, DEFCON 22.

[18] M. Gruteser, D. Grunwald, Enhancing location privacy in wireless LAN through disposable interface identifiers: A quantitative analysis, Mob. Netw. Appl. 10 (3) (2005) 315–325.

[19] R. Flickenger, S. Okay, E. Pietrosemoli, M. Zennaro, C. Fonda, Very long distance Wi-Fi networks, in: Networked systems for developing regions—NSDR'08, ACM Press, New York, New York, USA, 2008, p. 6.

[20] N. Lawson, Side-channel attacks on cryptographic software, IEEE Secur. Privacy Mag. 7 (6) (2009) 65–68.

[21] P.C. Kocher, J. Jaffe, B. Jun, Differential power analysis, in: CRYPTO'99—International Cryptology Conference on Advances in Cryptology, Springer-Verlag, 1999, pp. 388–397.

[22] P.C. Kocher, Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems, in: CRYPTO'96—International Cryptology Conference on Advances in Cryptology, Springer-Verlag, 1996, pp. 104–113.

[23] F. Monrose, M.K. Reiter, S. Wetzel, Password hardening based on keystroke dynamics, in: Computer and Communications Security, 1999, pp. 73–82.

[24] A. Panchenko, L. Niessen, A. Zinnen, T. Engel, Website fingerprinting in onion routing based anonymization networks, in: Workshop on Privacy in the Electronic Society—WPES'11, ACM Press, 2011, p. 103.

[25] T. Wang, X. Cai, R. Nithyanand, R. Johnson, I. Goldberg, Effective attacks and provable defenses for website fingerprinting, in: 23rd USENIX Security Symposium, USENIX Security 14, 2014, pp. 143–157.

[26] S. Chen, R. Wang, X. Wang, K. Zhang, Side-channel leaks in web applications: A reality today, a challenge tomorrow, in: Symposium on Security and Privacy, IEEE, 2010, pp. 191–206.

[27] A. Iacovazzi, A. Baiocchi, L. Bettini, What are you Googling?—Inferring search type information through a statistical classifier, in: Global Communications, IEEE, 2013, pp. 747–753.

[28] T. Stöber, M. Frank, J. Schmitt, I. Martinovic, Who do you sync you are? in: Security and Privacy in Wireless and Mobile Networks, (WiSEC), ACM Press, New York, USA, 2013, p. 7.

[29] F. Zhang, W. He, X. Liu, P.G. Bridges, Inferring users' online activities through traffic analysis, in: ACM conference on Wireless network security—WiSec'11, ACM Press, 2011, p. 59.

[30] J.S. Atkinson, O. Adetoye, M. Rio, J.E. Mitchell, G. Matich, Your WiFi is leaking: Inferring user behaviour, encryption irrelevant, in: Wireless Communications and Networking Conference, IEEE, 2013, pp. 1097–1102.

[31] J.S. Atkinson, J.E. Mitchell, M. Rio, G. Matich, Your WiFi is leaking—ignoring encryption, using histograms to remotely detect skype traffic, in: Military Communications Conference, IEEE, 2014, pp. 40–45.

[32] E.L. Quinn, Privacy and the new energy infrastructure, Soc. Sci. Res. Netw. (2009) 1995–2008.

[33] P. McDaniel, S. McLaughlin, Security and privacy challenges in the smart grid, Secur. Privacy Mag. 7 (3) (2009) 75–77.

[34] A.M. White, A.R. Matthews, K.Z. Snow, F. Monrose, Phonotactic reconstruction of encrypted VoIP conversations: Hookt on fon-iks, in: Symposium on Security and Privacy, IEEE, 2011, pp. 3–18.

[35] T.S. Saponas, J. Lester, C. Hartung, T. Kohno, S. Agarwal, Devices That Tell On You: Privacy Trends in Consumer Ubiquitous Computing, USENIX Security, 2007, pp. 55–70.

[36] A. Hadjittofis, WiFinespect Play Store Page, 2014. https://play.google.com/store/apps/details?id=uk.co.opticiancms.wifiprobe.

[37] TCPDump Team, TCPDUMP / LIBPCAP Repository, 2012. http://www.tcpdump.org/#documentation.

[38] Wireshark Foundation, About Wireshark, 2012. http://www.wireshark.org/about.html.

[39] W. Li, R. Mok, D. Wu, R. Chang, On the accuracy of smartphone-based mobile network measurement, in: Computer Communications, INFOCOM, 2015 IEEE Conference on, 2015, pp. 370–378.

[40] S. Shakkottai, R. Srikant, N. Brownlee, A. Broido, K. Claffy, The rtt distribution of tcp flows in the Internet and its impact on tcp-based flow control, 2004.

[41] MIT CSAIL, Project Sikuli Homepage, 2012. http://www.sikuli.org/.

[42] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2013.

[43] A. Lim, L. Breiman, A. Cutler, bigrf: Big Random Forests: Classification and Regression Forests for Large Data Sets, 2013.

[44] L. Breiman, Random Forests, Mach. Learn. 45 (1) (2001) 5–32.

[45] L. Breiman, Random Forests, 2013. http://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm.

[46] S. Schulz, V. Varadharajan, A.-R. Sadeghi, The Silence of the LANs: Efficient Leakage Resilience for IPsec VPNs, Transactions on Information Forensics and Security 9 (2) (2014) 221–232.

[47] M. Perry, Experimental Defense for Website Traffic Fingerprinting, Tech. rep., Tor Project, 2011.

[48] Twitter Inc., Twitter for Business—Targeting, Dec. 2015. https://business.twitter.com/solutions/targeting.

[49] Tor Project, Installing Tor on Android, 2015. https://www.torproject.org/docs/android.html.en.

[50] I. Ruiz, M. Nagappan, B. Adams, T. Berger, S. Dienst, A. Hassan, On Ad library updates in android apps, in: IEEE Software, IEEE, 2014.

[51] S. Dai, A. Tongaonkar, X. Wang, A. Nucci, D. Song, NetworkProfiler: Towards automatic fingerprinting of Android apps, INFOCOM, 2013, pp. 809–817.

John S Atkinson is a Researcher in Security & Crime Science specialising in Mobile Device Privacy and Network Security at University College London. His research is primarily focused on the technicalities and feasibility of inferring user activity despite encryption being used as designed (and protections against the ability so). More broadly, he retains a strong interest in the forensic, legal, social and – critically – usability aspects of security technologies. Engagement about “cybercrime”, privacy, digital forensics, security, technology or any combination thereof is always welcomed.

John E Mitchell is Professor of Communications Systems Engineering in the UCL Department of Electronic and Electrical Engineering. His research is focused on optical and wireless access systems, considering optical fibre access, millimetre-wave radio access, as well as systems that combine these two areas, specifically to support mobile front/back haul systems. This work has been funded by the UK Engineering and Physical Sciences Research Council (EPSRC), the European Union (FP7, Marie-Curie), the UK Technology Strategy Board (TSB) and industry (including BT and Selex ES). The outputs of this work have been published widely, as well as resulting in involvement in the organisation of, or as an invited speaker at, a number of major international conferences. He is currently on secondment to the UCL Engineering Faculty, leading a major undergraduate curriculum development programme across the faculty. Dr Mitchell is a Chartered Engineer, Fellow of the Institution of Engineering and Technology and a Senior Member of the Institute of Electrical and Electronics Engineers.

Miguel Rio is a Senior Lecturer in Telecommunications and Computer Networks at the Department of Electronic and Electrical Engineering, University College London. He belongs to the Communications and Information Systems Group where they are actively designing many components of the Future Internet and help changing the way people communicate and access information in the 21st Century. He has a long term interest in Multicast, P2P Real-time delivery, Routing Resilience and Quality of Service, Internet Topologies and Generation Models, Network Monitoring and Measurement, and Congestion Control and Congestion Exposure.

George Matich is a Chief Technologist with Selex ES Limited a Finmeccanica Company in the UK, he responsibility for Network Sensing Systems and has worked in the Aerospace and Defence domain for the last 30 years. During his time in defence he has lead developments of operational systems in communication, radio navigation and network sensing areas, where he has had responsibility for signal processing and systems engineering. George has also been involved in several UK CYBER initiatives and programmes and presented technical papers at several major international conferences. For the past ten years his responsibilities in Selex ES have also included managing company relationships with academia and specialist SME companies to ensure the maturation and acquisition of under-pinning research to support the company’s business needs. Work with academia has included the industrial supervision of 2 Ph.D. students and a number of M.Sc. projects at several UK Universities.