A Study of Third-Party Tracking by Mobile Apps in the Wild

University of Washington Technical Report UW-CSE-12-03-01

Seungyeop Han
syhan@cs.washington.edu
University of Washington

Jaeyeon Jung
jjung@microsoft.com
Microsoft Research

David Wetherall
djw@cs.washington.edu
University of Washington

ABSTRACT

We report the first field study of real-world tracking via mobile apps in which we measured how 20 participants were tracked over three weeks as they exercised their Android smartphone apps. We instrumented the phones with dynamic taint tracking to record communications that exposed identifying information, and inspected web cookie databases. We find that 36% of the sites (655 out of 1824) that our study apps are programmed to contact are tracking users. Of these sites, 37% track users with persistent identifiers (mostly AndroidId and IMEI) derived from an identifying string unique to the user's device. This is privacy risk as these IDs are long-lived and enable cross-application and cross-site profiling of the user; they are often sent without encryption and with geo-location too. Advertising and analytics services are widely used, being embedded in 57% of apps and tracking every single participant in our study. Most of these sites are heavy trackers: 25% of their tracking is done with persistent identifiers and geo-location is gathered by half of the top 10 advertisers. To one participant's surprise, over 600 geo-coordinates were exposed to a third-party advertising site during the study. Our participants expected to be tracked but wanted greater transparency and the ability to opt out in cases when there was no perceived value. To let them opt out, we prototype a privacy control that selectively shadows access to identifiers and other sensitive personal data by third-party components.

1. INTRODUCTION

As with the web, a rich tracking ecosystem is developing around mobile applications. Developer sites may track the progress of their users across sessions. Advertising companies such as AdMob help developers to deliver targeted ads to mobile users. And analytics companies such as Flurry Analytics provide libraries that app developers include to gather a variety of usage information to improve their products.

However, tracking on mobile phones is less well understood and potentially more invasive than tracking on the web. On the web, the vast majority of tracking is performed using cookies that distinguish between first- and third-party communications[1]. Users can inspect and block third-party cookies at any time if they do not want unseen parties to monitor their activities. Tracking on mobiles may be done in a variety of ways (as we discovered) and cannot be controlled by the user beyond the decision of whether to install the application (and grant it the required permissions) in the first place.

The privacy threat is heightened on mobiles because they contain a wealth of sensor data and personal information that may be used to profile users. Such information may include where users are and have been, their phone numbers and contacts, call and email histories, photos, and calendar events. Previous research including our own has shown that some apps do collect and send this information to third parties [8, 13, 9, 21]. The problem is exacerbated because trackers may use long-lived identifiers such as the IMEI (a unique device identifier) that can be used to link the profiles built by different applications and trackers into a single, more comprehensive record of activity.

Our main contribution is to present the results of what we believe to be the first field study to measure how third parties track mobile users via their smartphone apps. We set out to learn what mechanisms are used for mobile tracking, what kind of parties beyond developers are tracking users, how much tracking is actually going on, and what kind of information is being collected by these trackers. To do so, we consensually monitored 20 study participants over three weeks as they ran apps on an instrumented Android smartphone as part of their daily routines. Our instrumentation uses dynamic taint analysis to track and record the exposure of sensitive information (such as phone number and location) across the network, and inspects `WebView` cookie databases built up during usage.

We find tracking to be widespread using per-application web cookies and persistent identifiers, including those that require *no* user permission (AndroidId) as well as the previously noted identifiers that do require permission (IMEI). Most advertisers track users, and 13% (36 out of 268) of the tracking sites that use persistent identifiers belong to third-

[1]Web tracking may also use methods such as Flash LSOs and cache Etags, but this is viewed as adversarial and is less common [20].

party advertising networks or analytics services. All twenty participants of our study were tracked by these third-party advertising and analytics sites through 42% (94 out of 223) of the applications that they used during a three week period. Half of the top ten observed advertisers also collect the location as part of their tracking, use persistent identifiers, and transmit data without encryption. For one participant, collectively 615 geo-coordinates were exposed to one advertising site through a casual game application, to the participant's surprise. While some advertisers track responsibly and we found none that collected further personal information, we conclude that mobile advertising and analytics carries needless risk for the user because of its heavy use of persistent identifiers, combination with data such as location, and lack of encryption.

A longer term goal of our research is to learn how non-technically savvy consumers of mobile apps view their privacy and to provide them with well-matched privacy controls. To this end, we interviewed our participants before and after the study. We learned that, while they expect to be tracked and understand why certain information would be sent to other parties, they want greater transparency. Some of our participants were rather surprised when we showed them what trackers had collected about their activities. All but one wanted a way to opt out of tracking in situations for which they perceived no real benefit, which is likely to be the case for the more intrusive third parties. However, with the existing Android architecture, users are unable to opt out of tracking since granting permissions to an app does not distinguish between the developer and third party components. To remedy this, we prototyped a new privacy control that selectively shadows resources. It can withhold permission from third-party components while preserving access for other application components. This idea appears promising, and we plan to develop and evaluate it further in future work.

The rest of this paper is organized as follows. In Section 2, we define the mobile tracking problem that we explore and describe the various tracking mechanisms that we found. We describe our instrumented Android system that tracks the trackers in Section 3. Our study and its results are described in Section 4, the main part of this paper. In Section 5, we summarize participant feedback and describe a new privacy control that is derived from it. Related work is presented in Section 6 and we conclude in Section 7.

2. THIRD-PARTY MOBILE TRACKING

In this paper, we characterize third-party mobile tracking as it is experienced by real users. We begin by describing the mechanisms that can be used for tracking, along with their privacy considerations.

2.1 Tracking Mechanisms

We define tracking to be the linking over time of the activity of a mobile phone user into a tracking profile. At a minimum, some form of client state is needed to uniquely identify the user. The minimal tracking profile is thus a series of observations of when a user ran a specific app. This minimal profile may be annotated with many kinds of additional information, e.g., the location of the user at certain times, the phone number, and even the user's personal content such as contacts or SMS messages.

On the web, tracking is commonly performed by using browser cookies as the linking identifier [20, 15]. However, for mobile tracking, the common practices are largely unknown. Thus as our first task, we surveyed the previous literature [9] and undertook our own investigation of the Android OS. We enumerated four possible tracking mechanisms that are available to Android applications. We discuss each tracking mechanism with respect to the availability, duration, and scope of identifiers.

Cookies in WebView. Android applications may run remote code on a `WebView` integrated with the application's view. In that case, remote sites can store cookies inside the cookie database of the `WebView`. The cookie database is not shared across applications. These cookies are governed by the same policy as browser cookies so the use of cookies is restricted by the sites of the same domain that are accessed through the application's `WebView`.

System IDs. Android OS makes a unique string, AndroidId, available to applications. AndroidId is a 64-bit number as a hex string, randomly generated at the first boot. Retrieving this string does not require any permissions and it could allow cross-application profiling by remote sites. AndroidID persists with the operating system.

Device IDs. Each phone has a unique device ID tied to the device hardware. In Android, `TelephonyManager.getDeviceId()` method returns the IMEI for GSM phones and the MEID or ESN for CDMA phones. These device IDs persist with device even if the device is flashed with a new system image. These device IDs can be accessed by any applications with the `READ_PHONE_STATE` permission thus can be used for tracking the user across applications. android.os.Build.SERIAL is available since Android 2.3 and it does not require any permissions.

SIM card IDs. Through the same `TelephonyManager` API, Android applications that have the `READ_PHONE_STATE` permission may access the phone number, IMSI, and ICCID numbers which are unique to the SIM card. These SIM card IDs persist even across devices as long as the user keeps the SIM card.

The above methods provide convenient ways for apps to access and store tracking identifiers. However, since apps are programs with access to their own storage (e.g., database or files), there is nothing to stop them from creating and storing custom tracking identifiers. In fact, our investigation found one tracker (Google Analytics) that does this.

2.2 Privacy Considerations

We found it useful to view mobile tracking through the lens of web tracking to understand its privacy considerations. Here, we do so using the privacy framework established for third-party web tracking [5] and highlight the heightened privacy risks of mobile tracking.

Lack of separation between first and third parties. On the web, third-party tracking refers to the practice by which users are tracked by websites that they do not directly visit [5]. Browsers are capable of distinguishing first-party sites (shown in the address bar) from third-party sites. This facilitates techniques such as the same-origin-policy [3] to limit cookie sharing based on domain names.

In mobile tracking, this distinction is lost. There is no operating system support with which to define third parties, so apps with permission to use the network may send to any destination without restriction. In our previous work, we found that many Android applications are compiled with code from advertising networks and analytics services [13]. Once compiled, the code from these third parties is endowed with the same permissions as the first-party code. This means that if the application has permission to access phone identifiers, presumably for a purpose intended by the user, then the third-party code is equally capable of accessing the phone identifiers. This capability is easily exploited by third parties to build a detailed profile of the user across many applications running on the same phone.

Despite the lack of a formal definition, we continue to consider third parties to be "sites the user does not directly visit". For example, communicating with the developer site such as Rovio for Angry Birds is a first-party task, while communicating with AdMob from the same application is a third party task. In practice, this division is more difficult to make than it sounds because of fetches that are part of user-manipulated content (e.g., images in email messages) and outsourced developer services (e.g., use of CDNs). Users might reasonably expect these interactions to be considered first party when they are essential to the primary purpose of the application. Given this diversity, we manually classify communicating parties to identify third parties as described in Section 4.2.

Lack of user privacy controls. Most browsers provide mechanisms that help users (or at least savvy users) see how they are being tracked and to opt-out of unwanted tracking, e.g., inspecting the cookie store, plug-ins [20], blocking third-party cookies [2], and HTTP options [16]. In sharp contrast, mobile users can neither see nor control how they are tracked. Android applications do not provide users with a mechanism to manage cookies in `WebView`. The system ID may be deleted using the factory reset option. However, exercising factory reset is costly as this will wipe out all other application data as well. The device IDs and SIM card IDs are protected by the `READ_PHONE_STATE` permission. However, this permission mixes access to basic phone context (such as whether the user is in a call) with identifiers that can be used for tracking. Further, Android applications employ an all-or-nothing permission model so there is no way to disable already granted permissions without uninstalling the application.

[2]Many trackers also provide their own opt-out systems by having users store their preference in special cookies [2].

Use of real-world names. Although arguably some of the identifiers discussed in the previous section are not meant to be used for tracking [1], previous studies [9, 13, 21, 7] have found that Android and iPhone applications share the device ID with third-party advertising networks and analytics services. Unlike web cookies, which are virtual identifiers, the device ID and SIM card IDs (such as phone number) are tied to real-world entities—a particular mobile phone that is closely associated with an individual. By their nature, these identifiers are long-lived and difficult to change. This means that mobile third-party tracking may be done with identifiers that allow activity to be linked to people over long periods of time.

Availability of Sensor Data. Finally, mobile activity differs from traditional browsing in the availability of sensor data such as location, images, and audio on mobile phones. This data may be collected and sent to third parties if the application has permission to access it. The combination of sensor data and real-world names means that tracking may create profiles that are highly personal for some individuals, e.g., daily movements or even a record of conversations.

3. TRACKING APP BEHAVIOR

Our tracking results are derived from measurements of app behavior collected on instrumented Android smartphones. In this section, we describe the AppLog system that we developed for this purpose, including its design goals, the kind of data that it collects, and highlights of its implementation that are relevant to the results in the next section.

3.1 Design Goals

We were guided by three design goals in building the AppLog system.

Observation of regular usage. Our study observes the regular activity of consenting participants over a relatively long period of time. To do so, our instrumentation must work with a wide variety of apps, and it must be lightweight enough that apps run without noticeable performance degradations. A lightweight system will also consume relatively little in the way of other mobile resources such as energy and network bandwidth, so that it is not necessary for the user to significantly alter their behavior during the study.

Since tracking is blended deeply into an application's binary, we were faced with the choice of instrumenting either every Android application or the system runtime in which these applications are executed. We chose the latter approach because we knew from our earlier experiences that we would be able to cover a large set of real applications without compatibility issues and with low runtime overheads. The cost of this choice is that we require participants to use mobiles that we provide with a custom OS.

Capture contextual information. Tracking as we have defined it refers to the collection of a profile of linked activity. However, to understand tracking, we determined that we would need to collect more than simply the profile that was communicated to external parties. The key missing element in our early explorations was context that is valuable to understand how and why tracking is occurring.

For example, Android applications can run in the background even when the user is not directly engaged with them. Consider an app that sends tracking identifiers with the current location to an external party. This application might be a location-based service (such as a map) that is being exercised by the user, or it might be a tracker that is operating unbeknown to the mobile user. These two scenarios are quite different in terms of their privacy expectations, but cannot be distinguished without contextual information given that the Android OS provides no ways for users to review which applications have accessed location information in their phones. Another example of contextual information we find useful is the use of encryption associated with network activity. This lets us determine how often the user's identifying and personal information is collected without proper protection.

Respect the privacy of study participants. We must respect the privacy of study participants who allow us to monitor tracking activity embedded in applications that they use in their everyday life. Since we collect rich contextual information, we should provide an easy way for participants to temporarily opt out from our monitoring during the study period. We should also minimize the collection of message payloads as they may contain sensitive information, yet find ways to detect when sensitive information is exposed to third parties.

3.2 Data Collected

We collect the following information.

Use of tracking identifiers. To discover which trackers use which mechanisms, we monitor the use of the system, device, and SIM card identifiers by applications, as well as the cookies in the WebView database and in the `sharedpreference` file. This monitoring lets us learn which identifiers are shared with which external parties.

Network communications. Tracking information is exported in network messages. We log the frequency and volume of communications by each app, including a record of each external server with which the app corresponds. Messages that contain tracking identifiers receive special attention: we log their header for later inspection.

Personal information. In addition to tracking identifiers, we monitor the transmission of several types of personal information that may be used as part of tracking profiles. This information includes location, contacts, phone number, calendar, and SMS messages.

Other context. Finally, we collect other context surrounding the state of the app that can help us understand tracking behaviors. This context includes the process state (e.g., background, foreground, just closed) as well as the encryption state of network traffic (e.g., SSL).

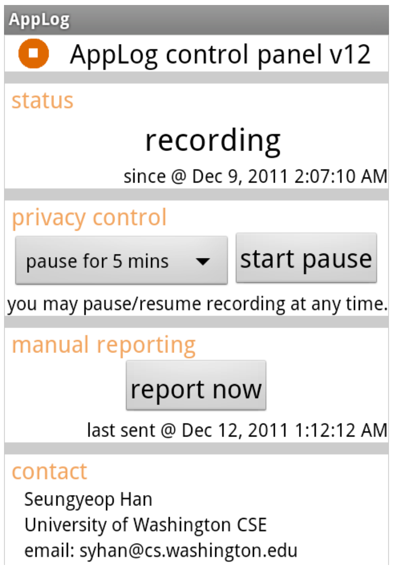
3.3 AppLog System

The AppLog system that captures the information described above consists of two parts: a modified Android OS and an Android application. The OS part is built upon the TaintDroid[3] system that dynamically tracks information flow at the Dalvik bytecode level. We extended this system to track new data types, notably the AndroidId, and to propagate taints over commonly used native functions, such as MD5 hashing. Among the identifiers that could be used for tracking (persistent IDs), sources for device identifier, phone number, ICCID, and AndroidId are tainted. IMSI is excluded from TaintDroid due to false positives [8]. The hardware Serial number is not tainted as it was found after the study had started. Finally, we note that TaintDroid does not track taint through binary applications components.

The Android application part provides the rest of the system. It collects records of the tracking identifiers that are sent over the network from the OS portion, and uses standard logging components of the Android OS to collect detailed information about applications' activities. Communication between the OS and the application relies on Android's lightweight logging system. When events to be collected occur, the OS writes a log entry. The application monitors the log to get the data.

The application is responsible for aggregating the various data sources, storing them into files, and shipping them over the network to our collection server. Conceptually it is straightforward, though in practice it is difficult to reliably collect data. Each record is timestamped, and annotated with the current location if it is available. However, we do not turn on the GPS if it is not already active to avoid a power drain, instead simply using the best known location. An example of a record for the exposure of sensitive information is:

[3]We use a version of TaintDroid based on Android 2.3 released in Aug. 2011.


AppLog

AppLog control panel v12

status

recording

since @ Dec 9, 2011 2:07:10 AM

privacy control

pause for 5 mins | start pause

you may pause/resume recording at any time.

manual reporting

report now

last sent @ Dec 12, 2011 1:12:12 AM

contact

Seungyeop Han
University of Washington CSE
email: syhan@cs.washington.edu

Figure 1: AppLog application screenshot.

- When: Dec. 12th, 20:18
- App: Facebook
- Destination: api-read.facebook.com
- Status: Foreground
- Type: Location
- Where: 47.653, -122.306
- Using SSL: Y
- Sent: 547 bytes

AppLog also includes a GUI that lets the mobile user see the state of and temporarily suspend data collection, if desired. It is shown in Figure 1. The GUI did not let study participants view the data that was collected about their usage to avoid influencing their behavior.

On the server, the shipped files are parsed and stored into a database for later analysis.

4. MOBILE TRACKING STUDY

In this section we report on our study of how real consumers are being tracked while using smartphone applications. We first describe the setup for our study, then analyze the tracking measurements that we collected. We focus on tracking for advertising and analytics, including privacy risks in terms of the size of tracking profiles.

4.1 Study Method and Setup

We recruited participants through online and offline flyers posted at local coffee shops, Craigslist, the researchers' personal social networks, and classifieds of the school's daily newspaper. Twenty people participated in our study (9 females, 11 males, aged 18 - 41). Although many of our participants were drawn from the school (10 students, 3 research assistants, but only 2 from computer science), the rest represented a mix of professions including web designer, artist, cook, and home maker. We screened interested people to select moderate to heavy mobile phone users. Each participant was an everyday user of many downloaded Android apps at multiple locations. All participants had a personal mobile phone that they did not share with other users, had been using Android for at least one month, and had at least 200 MB data plans with T-Mobile—17 had unlimited data plans, and 3 had access to Wi-Fi at most times.

At the beginning of the study, we moved the SIM card and copied the set of apps from each participant's mobile phone to an experimental Nexus S mobile phone that we provided. This mobile ran a custom Android 2.3 OS with our AppLog system: it collected application and network usage measurements throughout operation and uploaded them in the background to our server as described in Section 3. The participants used the experimental mobiles as their primary phone during our study, as a drop-in replacement for their own mobile; we copied apps and data such as photos off the experimental phones and back to personal phones at the end of the study to encourage the participants to use the experimental phone as they would their personal phone.

We collected data from each participant for a minimum of 3 weeks from November to December 2011. During the study, the participants were able to pause data collection if they desired for privacy reasons, though this option was rarely exercised. We were able to collect measurements for the vast majority of the study period, except for a small number of occasions when we lost data for the remainder of a day due to a crash of the AppLog application component. We also discovered and corrected a bug in logging the size of SSL-protected communications early in our study. It affected 7.1% of the network data points, and we exclude them from our bandwidth results.

The uploaded measurements plus the cookie databases built up during usage are the primary sources of data for our analysis. Our results that follow in the next sections give lower bounds for tracking activity because we conservatively assume there was no tracking when faced with data collection or analysis limitations (e.g., missing data, or interpreting application cookies).

At the end of the study, we interviewed the participants about their experiences to understand how they viewed tracking and related privacy issues. This qualitative feedback is discussed in Section 5. All but one participant reported that they did not experience any unusual problems such as slowness or shorter battery life with the study phone that could have prevented them from using a usual set of applications during the study period. The one participant who noticed significant performance degradation when switched to the study phone still ran over fifteen different applications during the study period.

4.2 Trackers Counted & Categorized

We first present overall statistics for the tracking sites that we identified from the measured data. We then show how these tracking sites can be categorized for further analysis.

Table 1: Overall tracker statistics.

	#sites	#domains	#apps
Persistent IDs	242	93	152
Web cookies	434	150	72
Trackers total	655	224	168
Dataset total	1824	591	223

Dataset. Over the three week study and across all 20 participants, we collected usage information for 223 distinct applications that communicated with over 5000 different destinations. We began our analysis by excluding web browser and email client activity from our dataset, as it is not the focus of our study. These apps tend to contact a large number of remote servers as directed by users or user content (rather than directed by the developer) and their communications are simply tracked in the same manner as non-mobile web usage. We also exclude parts of the Android operating system that show up in our data collection as applications, e.g., system processes used to sync data.

After excluding these cases, we convert the destinations into sites and domains to approximately represent the servers and companies with which the apps communicated. Sites are simply fully-qualified DNS names. We use sites rather than IP addresses because they allow us to more readily link network usage to web cookies; a small number of destinations (151) are excluded because they have no corresponding DNS name. To obtain domains, we merge sites that have the same top two levels of naming (e.g., amazon.com). This process leaves us with 1824 sites that come from 591 domains.

The remaining set of apps and their communications to sites and domains form the basic dataset that we analyze in this section. As a first analysis, we look for the presence of tracking. To do so, we divide tracking into two broad classes. The first class is the use of persistent identifiers, e.g., AndroidId. The exposure of these identifiers is directly recorded by AppLog. The second class is the use of web cookies, whose use is identified by post-study analysis of the participant mobiles. Each application with a `WebView` component maintains a `webview.db` database that contains a `cookies` table. We pull these databases and match the domain attribute of each cookie entry against the list of sites. An exact match is not needed: the cookie domain may be a parent of the site, e.g., cookies for `.foo.com` will be sent to the site `a.foo.com`.

Overall Tracker Statistics. We find that tracking is widely distributed across sites, domains, and apps. Table 1 gives the breakdown. 36% (655 out of 1824) of the sites with which monitored applications communicated tracked users either using web cookies or persistent identifiers. Among these tracking sites, 37% (242 out of 655) tracked users with persistent identifiers. AndroidId (74 domains) was the most widely used persistent identifier, followed by IMEI (52 domains). Other identifiers were used less commonly: phone number (15 domains) and ICCID (2 domains). We find the use of persistent identifiers to be surprisingly high given that they pose a potential privacy risk.

There is likely further tracking that we do not discover because we were not able to measure tracking application-generated unique IDs stored in their own storage. Unlike web cookies, the existence of entries in other databases does not correlate well with exposure. Similarly, while the `WebView` component is popular, apps may develop their own cookie support instead of using it. One important exception we discovered is Google Analytics, which manages its own cookies as part of the application in which it is hosted. We included Google Analytics in our tracking statistics above, but may have missed other cases of custom tracking in our manual checks.

Overall Tracking by Participants. The tracking statistics given above are aggregated over all participants and all kinds of activity. Our next analysis was to separate tracking by participant to see how much it varied.

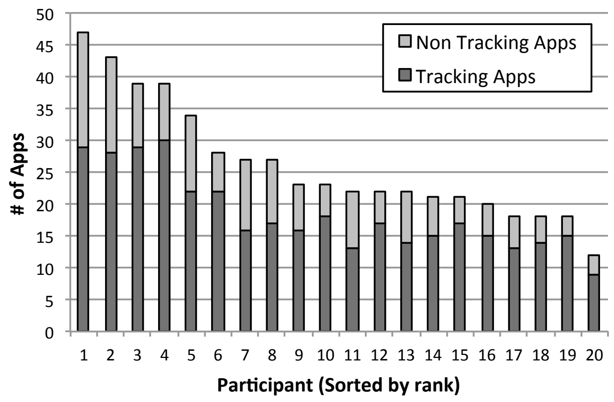


Figure 2: The number of applications used by each participant.

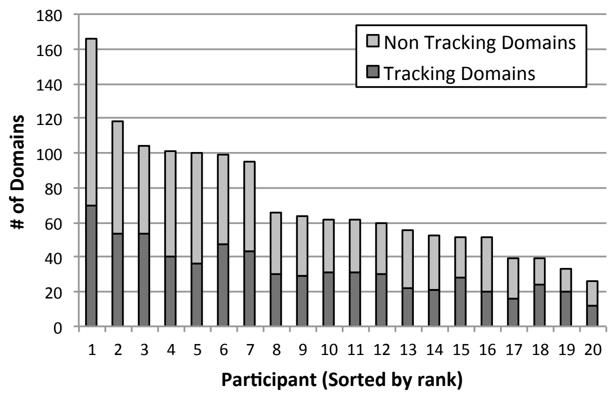


Figure 3: The number of domains contacted by each participant.

Table 2: Category of observed communications

	Persistent IDs			Web cookies			Total tracking			Total		
Category	sites	doms	apps	sites	doms	apps	sites	doms	apps	sites	doms	apps
Primary	123	46	51	92	24	33	208	63	76	518	167	156
Advertising	36	21	74	120	38	37	149	52	82	264	105	93
Content Server	61	14	35	190	79	27	246	90	49	930	281	117
Analytics	6	5	34	11	8	12	17	12	41	37	26	81
API	14	10	48	19	8	23	31	14	55	66	33	71
Don't Know	2	2	2	2	2	2	4	4	4	9	9	10
Total	242	93	152	434	150	72	655	224	168	1824	591	223

The number of applications used by each participant is shown in Figure 6. The applications are broken into those that embeded tracked activity and those that did not. The participants used from 12 to 47 different applications, and each participant used 26 applications on average. We see that tracking is embedded in most of the apps that are used by all participants; it is not concentrated on a fraction of the participants. This tracking is used for many purposes, including first party tracking by developers, and we will break down the classes of tracking shortly.

A different view of overall tracking is given by the number of domains with which each participant communicated. It is shown in Figure 3. Compared to apps, there is more variation across participants as some apps contact many domains (even excluding web browser and email clients) while others do not. The participants contacted from 26 to 166 domains. Tracking activity can also be seen to be concentrated in a smaller fraction of the domains, i.e., most apps embed tracking but most of the domains they contact do not perform tracking. Nonetheless, all of our participants are tracked by multiple domains.

Categories of Trackers. All tracking is not alike: the purpose for which it is performed is a key factor in how users perceive it. For example, tracking that is performed by the developer as part of the operation of the app (e.g., identifying the mobile to game servers) is likely to be considered benign by most users (as many apps use some sort of account with the developer). As such, it is not a focus of our study. On the other hand, tracking that is performed by an unknown third party analytics company that operates unseen by the user is more likely to be a privacy concern.

To let us focus on this kind of third party tracking, we classify communications in our dataset according to their likely purpose. Unfortunately, there is no widely applicable set of rules such as first versus third party in web browsing, nor is there operating system support for discerning network usage as all communications occur undifferentiated within a single protection domain. Instead, we turn to manual classification: we probe the site that is contacted, search for it on the Web, use WHOIS registration data, and in a small number of cases inspect decompiled application code. This investigation typically reveals the likely purpose of the communication, and while it is necessarily inexact we found it to be highly informative for understanding what applications are doing when they use the network.

The breakdown of communications by category is given in Table 2. The `Primary` category is for the app communicating with the developers' own servers. The `Advertising` and `Analytics` categories are the main third party communications that we identify. The `Content Server` category captures communications that deliver developer-selected content to the app. An example is a CDN download. This category might be considered to be a third party interaction, as it is typically a separate business than the developer, or a first party interaction, in the sense that it is the use of a service by the developer to gather content required for the app to run. Similarly, the `API` category covers other kinds of cloud services that developers commonly make use of through published APIs, including maps, weather, authentication, and social plug-ins. Finally, for a small fraction of the communications (`Don't Know`) we are unable to clearly discern the purpose.

Tracking can be seen across all of our categories. Unsurprisingly, the bulk of apps (but not all of them) make use of `Primary` communications, and these communications are often tracked. While fewer apps include `Advertising` communications, these apps are more likely to be tracked by a more concentrated set of domains. `Analytics` is present, but comprises a smaller fraction of the communications and tracking than `Advertising`. Finally, there is a good amount of `Content Server` and `API` communications as apps make use of various services in the cloud.

Breakdown using Categories. By assigning communications to categories, we can divide the overall behavior of apps. This breakdown is shown for the 10 most popular apps across our participants in Figure 4. The figure reveals the large degree of diversity in app behavior. Some apps such as Evernote are solely focused on their primary task (of cross-device access to personal notes). Other apps such as Pandora have a well-developed mix of behavior, including the use of CDNs and cloud APIs (e.g., for maps) as well as advertising and analytics services. In terms of network and

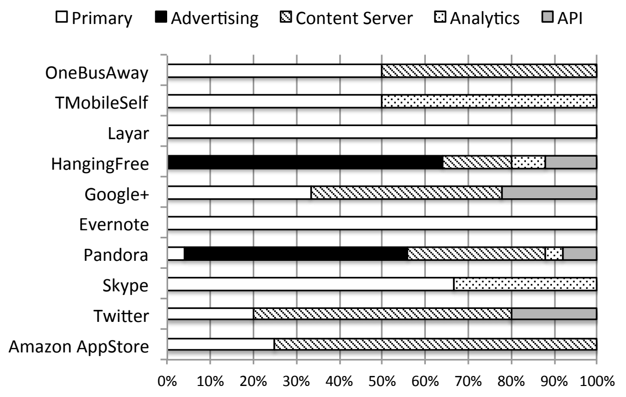


Figure 4: Breakdown of domains with which each app communicated

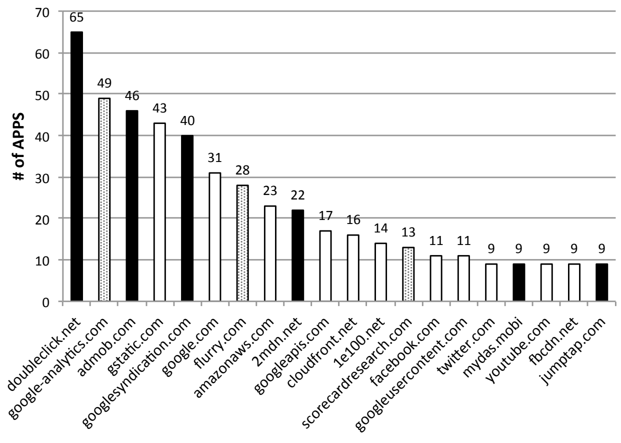


Figure 5: Popular target domains used in mobile applications.

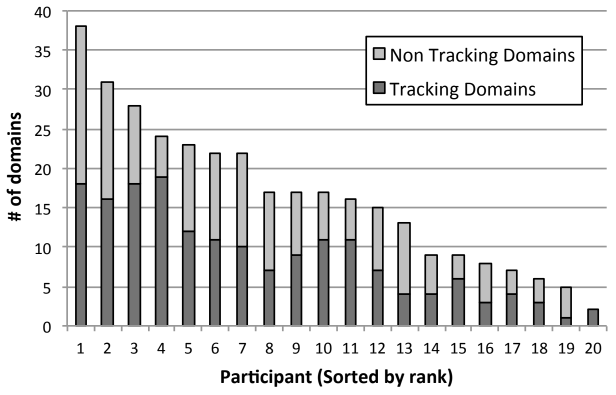


Figure 6: The number of advertising domains contacted by each participant.

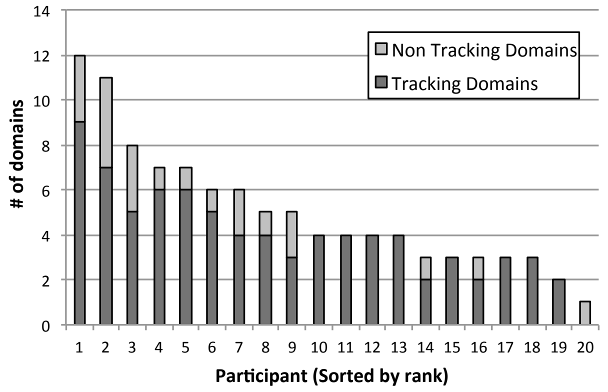


Figure 7: The number of analytics domains contacted by each participant.

tracking behavior, there is no meaningful notion of a "typical" app.

We can also look at the most popular domains to see what kind of behavior they represent. Figure 5 shows the number of applications across our participants that communicated with the 20 most popular domains. This domains in this figure are categorized using the same legend as in Figure 4. Some of the domains are very popular and used by a large fraction of apps, with Google domains for advertising and analytics being the most obvious example. Most but not all of the popular domains perform tracking. A good example of a popular site that does not track is `gstatic.com` (Google). It delivers static content such as Javascript to mobiles. Of the sites that do track, most of them use persistent identifiers that enable long-lived, cross-app tracking.

4.3 Advertising & Analytics Tracking

Advertising and analytics are the categories of tracking we have identified that are most likely to be associated with privacy concerns by users. In this section we delve into them more deeply to learn how they operate.

A&A by Participants. To begin, we give a breakdown of the number of advertising and analytics domains that are contacted by each participant in Figures 6 & 7. Advertising domains outnumber analytics domains, and there are a surprisingly large number of domains. Most participants are communicating with more than a dozen different domains, and most of the domains are tracking users. The amount of traffic used for advertising and analytics can also be quite large as shown in Figure 8. In the case of four participants, advertising traffic consumed over 5 MB bandwidth during a three week period.

Popular A&A Domains. While there are many different advertising and analytics companies, a small number of companies are widely embedded and receive traffic from many of the apps on the participants' mobiles. This pervasive embedding raises the concern of cross-application tracking. Table 3 presents data on the advertising and ana-

Table 3: Top 10 advertising and top 10 analytics domains

Advertiser	Apps	Participants	Tracking Identifier	Location	SSL	Cross-App
doubleclick.net	65	20	md5(AndroidId)/Cookie		Sometimes	×
admob.com	46	19	md5(AndroidId)	Y	Never	×
googlesyndication.com	40	19	N/A		Sometimes	
2mdn.net	22	14	N/A		Never	
mydas.mobi	9	10	IMEI	Y	Never	×
jumptap.com	9	9	IMEI	Y	Never	×
atdmt.com	8	9	Cookie		Sometimes	
admarvel.com	8	12	AndroidId and IMEI	Y	Never	×
inmobi.com	7	9	AndroidId	Y	Never	×
mojiva.com	7	8	N/A		Never	

Analytics	Apps	Participants	Tracking Identifier	Location	SSL	Cross-App
google-analytics.com	49	18	Cookie (separate DB)		Sometimes	
flurry.com	28	17	AndroidId	Y	Sometimes	×
scorecardresearch.com	13	14	md5(Serial, salt)†		Sometimes	
quantserve.com	7	7	Cookie		Sometimes	
medialytics.com	6	6	md5(AndroidId)	Y	Never	×
imrworldwide.com	3	3	Cookie		Never	
statcounter.com	2	2	Cookie		Never	
localytics.com	2	5	SHA256(AndroidId)		Never	×
chartbeat.net	2	2	Cookie		Never	
bluekai.com	2	2	N/A		Never	

†Uses AndroidId instead of android.os.Build.SERIAL if the system is running a version prior to Android 2.3.

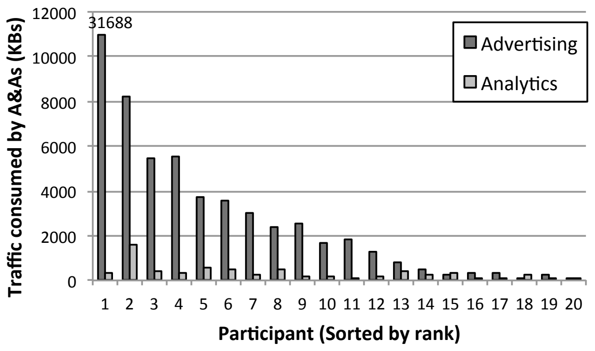


Figure 8: Traffic consumed by advertising for each participant.

lytics domains that were most frequently contacted by apps.

Nearly all these domains are used by most of the participants, and many are used across a large number of apps. Of the domains, all but four perform tracking, and the majority of the trackers are based on persistent identifiers. However, the use of persistent identifiers does alone not imply that cross-application tracking is possible. This is because an app might derive a unique but otherwise unlinkable identifier from the persistent identifier by hashing it together with some salt. To investigate this question, we decompiled apps that used the advertising and analytics domains and inspected how they made use of the persistent identifiers. Domains capable of cross-application tracking are indicated in the tables.

We found that in half of the cases the trackers do use IMEI or AndroidId directly, and they most often send this identifier in plaintext so that any other party on the network (e.g., a nearby WiFi user) can intercept it. The other trackers derive an identifier from the persistent identifier, but most often in a reproducible manner, e.g., by taking the MD5 hash. This is slightly helpful because it makes cross-application tracking somewhat less obvious, but does not prevent it because it is a simple matter to check whether an identifier is the MD5 hash of another identifier. A much better method from the viewpoint of privacy is to hash the persistent identifier with some salt so that a new, unlinkable identifier is produced. We find one analytics provider, `scorecardresearch.com`, that follows this practice.

Half of the popular advertising domains also collect the location as part of their tracking, while location tracking is less common for analytics companies. Unfortunately, the advertisers that collect tracking are the same ones that use persistent identifiers and they further send both pieces of information across the network without protecting it from other parties with encryption. It is hard to imagine a worse combination of choices from a privacy standpoint. These advertisers are `admob.com`, `mydas.mobi`, `jumptap.com`, and `admarvel.com`.

Our measurement also reveals that in one case, 615 geo-coordinates of one participant were exposed to `admob.com` along with AndroidId in most cases (480 out of 615) through a game application that the participant frequently interacted with during the study period. As we discuss in Section 5.1, the participant found this practice surprising as the game application had no use for location. Although this is an outlier, advertising and analytics sites collected at least 38 geo-coordinates from 40% (8 out of 20) of the participants over three weeks. This practice is likely to be unsettling to many users.

Besides location, we recorded no instances in which the other types of sensitive personal data that AppLog monitored were collected by advertising and analytics sites. Conversely, AppLog did record many instances in which these data types including contacts, microphone data, and photos were transmitted off to `Primary` sites for (presumably) legitimate reasons.

4.4 Tracking Profiles

Privacy concerns for mobile tracking center on the use of persistent identifiers that allow tracked information to be linked across applications and even tracking companies. In this section, we develop our analysis of the top 10 advertisers and analytics trackers in our dataset to explore these risks.

We gauge privacy risks in terms of the size of tracking profiles that are built by each tracking domain. Each tracker collects tracking events for a particular user from each app that embeds it. The way that trackers combine events into profiles depends on the tracking identifier that is used. If it is a persistent identifier (e.g., IMEI or AndroidID) then usage events for a given user from different apps can be combined into a single profile. If it is a web cookie then they cannot; the tracker cannot distinguish one user running both apps from two users each running one of the apps.

Tracking profiles should be strictly equal or larger under the persistent identifier model than under the web cookie model. Tracking as we have measured it falls inbetween these models because some domains use persistent identifiers for tracking while some use web cookies. There is also one further model that we consider: the merger model. A risk with persistent identifiers is that separate profiles may become linked over time, say when two different tracking domains that are owned by the same company join their profiles by using the same persistent identifier as a key. Under the merger model we consider what would happen to the tracking profiles we measured in the worst case that all tracking companies combine their profiles where persistent identifiers make it possible.

Tracking profile size for the web cookie model (`Web`), as we measured it (`Measured`), and for the merger model (`Merger`) is shown for each participant in Figure 9. For each model, we selected the largest profile size to show the highest level of tracking that was (or would be) encountered. Given that persistent identifiers are frequently used for tracking, the measured tracking profile is on average 1.4 times larger than if trackers had used web cookies. In the worst case, for participant ID 14, the measured tracking profile is 4 times bigger than the profile under the web model.

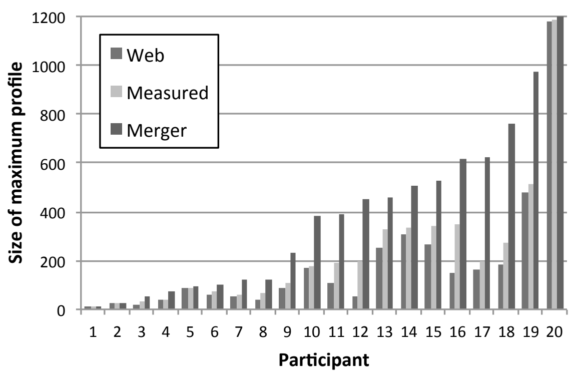


Figure 9: Largest tracking profile size under different models. What we measured (`Measured`) is compared with the web model (`Web`), which keeps separate per app per domain profiles, and the merger model (`Merger`), which aggregates per app per domain protocols that use the same persistent identifier for tracking.

We find that the hypothetical merger model would allow these advertisers and analytics companies to grow the size of their tracking profiles by an average 1.8 times. In the worst case, for participant ID 19, the profile grows by a factor of 3.1. The inflation is somewhat smaller than expected, but not surprising given that we only looked at twenty domains and the monitoring period was only three weeks. The potential gain will only grow over time and as users change their working set of apps. Nonetheless, this modeling demonstrates the heightened privacy threat of the popular advertising and analytics companies that track users with persistent identifiers.

5. A NEW PRIVACY CONTROL

Our field study provided us a unique opportunity to communicate the insights learned from data analysis back to participants. We first present a qualitative discussion of participant feedback as to how they view tracking and what privacy controls that they desire to manage related privacy risks. We then present an initial prototype of a privacy control to let users opt out from unwanted third-party tracking.

5.1 Participants' Reactions to Tracking

After running an instrumented study phone for three weeks, participants returned to our lab for a one hour in-person session. During the session, they completed a questionnaire about the usage of the study phone and were interviewed about their study experience and privacy concerns toward personal data exposure, especially to third party companies. This section presents a few highlights from the preliminary

Table 4: Views on data collection by third parties

Description	%
The application should never share my personal data with third parties without my explicit consent (i.e., opt-in choice)	45%
The application may share my personal data with third parties but should give me a choice to opt-out	50%
The application may share my personal data with third parties and I would not want to be bothered with a choice neither to opt-in nor opt-out	5%

analysis of the interviews with participants and the design guidelines that shaped our prototype privacy control.

The first set of interview questions focuses on assessing whether participants have a reasonable understanding of how often the applications that they commonly use collect their personal data, and among those that do which applications share data with third parties. We first quizzed participants about ten applications that were recorded by AppLog then showed them the measurement data to compare their expectation with the applications' actual behavior. Interestingly, participants have a good understanding of how applications work and why they might have collected personal data such as location, contacts, and phone number. However, quite a few participants were unsettled when they found out that some applications further shared information with third parties such as advertising and analytics companies, especially when these are not primary applications that they interact with on a regular basis.

After having a discussion about applications' data sharing with third parties and possible reasons for why such sharing may happen (e.g., targeted advertising, market research, reselling data), we asked participants to choose the statement that best describes how they feel about current practices. Table 4 shows the breakdown of participants' response. As the table shows, all but one participants wished to have an option to opt-in or opt-out.

To better understand the desire to control how applications share personal with third parties, we presented a list of possible reasons with a seven point Likert scale from strongly agree (7) to strongly disagree (1). These reasons were adapted from a seminal study [17] that deeply investigated people's attitudes and privacy concerns about behavioral advertising practices. The results are shown in Table 5. 80% of participants agreed that they would watch how they install and use applications that are known to share personal data with third parties. One participant specifically singled out one application that had shared location data with third parties evidenced by our measurement and mentioned that s/he would uninstall it from the phone. Another participant mentioned that s/he would not have installed two free game applications that had used IMEI for third-party tracking.

We make two observations from the interviews. First, a mechanism to help people monitor third-party tracking behavior can be beneficial to users. Repeatedly during the interview, participants wanted to know more details about the third party companies that tracked their information. Some asked for the list of these third party companies for their own investigation. This attitude is also reflected in Table 5 which shows that 90% of the participants strongly disagree or disagree that they do not care if applications share personal data with third parties. Only one person agreed that they do not care and one person was neutral.

Table 5: Mean Likert scores to accept or reject third-party tracking (strongly agree = 7, strongly disagree = 1)

Description	Mean	Agree	Disagree
Someone keeping track of my activities while using mobile applications is invasive.	4.65	60%	30%
I would watch how I install/use mobile applications more carefully if I knew applications were sharing my personal data with third parties.	5.85	80%	5%
I do not care if applications share my personal data with third parties.	2.15	5%	90%

Second, a mechanism to help people selectively block third-party tracking through mobile applications is strongly desired by users. Most participants understand the advertising supported ecosystem. As one participant put it *"I recognize that advertising is part of the price that I pay for getting something free"*. However, they clearly express the desire to know which third-parties are tracking their information and to have the capability to opt-out or opt-in from third-party tracking depending on the value that they get from the application. For instance, after seeing the popular Angry Birds application had shared information with multiple third parties including location, one participant mentioned s/he would opt out from third-party tracking, explaining that *"An app like a game or something like that didn't really need that (location) information and had no use for it other than for advertising purposes, I would not want it to share..."* .

5.2 Initial Prototype

Encouraged by participant feedback, we set out to provide a way to opt out of third-party tracking. Our discussions emphasized to us the lack of privacy controls for mobile tracking compared to the web. On the web, there is a browser-level distinction between first- and third-party communications, various browser and add-on mechanisms to manage how cookies may be used for tracking [20], and proposed HTTP level support in the form of Do-Not-Track [11]. On mobiles, third party code, e.g., for advertising and analytics, runs in the same protection domain and with the same privileges as the rest of the application. Thus, if the app has access to location (or other personal information) for its own purpose then this information can always be re-used by third-party code.

To opt-out of tracking, we need to differentiate parts of the app so that a third-party tracking component can be denied access to tracking information while the rest of the app can function as usual. This differentiation is possible because of the structure of many apps: developers typically take a library of third-party tracking code that is written separately from the rest of the app and then include it. Note that this differentiation is a different goal than simply blocking communications with known advertising domains (e.g., AdFree). Blocking breaks network operations, while we want to allow operations but disable tracking. While there is other work on privacy controls for mobiles, e.g., MockDroid [6] and our earlier AppFence work based on blocking taint [13], such work treats apps as monolithic.

5.2.1 Design and Implementation

Design. Given that apps are comprised of separate packages, we use the package structure to define protection boundaries. We define some packages that represent third-party advertising and analytics as restricted and draw on Java concepts for separating the privilege of code components running in the same process [22]. When sensitive information such as the IMEI or location is accessed (given that the app already has sufficient permissions) we use the call stack to determine whether the access originated from a restricted package. If so, we shadow the access and return information that does not support invasive tracking, e.g., a salted IMEI or a false location. We refer this mechanism to selective data shadowing to distinguish from monolithic data shadowing in previous work.

The above mechanism is sufficient to let users opt out of third-party tracking with persistent identifiers. However, it does not address web cookie tracking. Fortunately, web cookies can be deleted for a domain that is marked as restricted every time an app is run. This causes the app to generate new cookies (as in a first time of use) that are not normally linked to previous cookies (given that we can shadow persistent identifiers).

Implementation. We modified our custom AppLog OS to insert check code at all API sites that access information that may be restricted (e.g., IMEI, AndroidId, location). Using Java's stack introspection facilities, we can check whether the access has come via a restricted package. Note that it would be also possible to modify app binaries to insert these checks; OS modification (and taint propagation) is not necessary, though it was convenient for us. In either case, the runtime overhead is very low because there are few calls to the restricted API sites and hence few checks.

Figure 10 shows an example of two different components in the `Fruit Ninja` app that access IMEI. From the call stack, one call is identified as coming via `com.mobclix`. Its

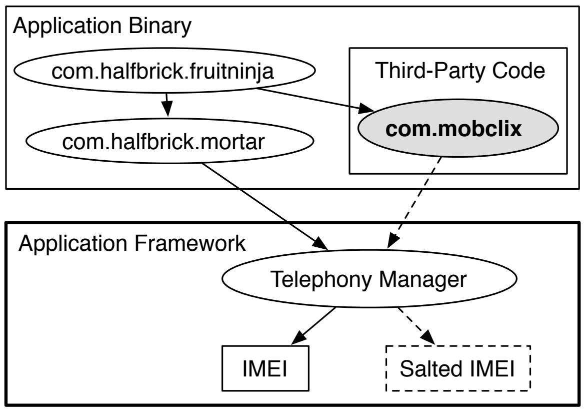


Figure 10: Two partial call stacks to the instrumented `Telephony Manager`: our privacy control layer shields access to the protected resource, `IMEI`, by third-party code using stack introspection.

access to the IMEI is restricted by returning a salted IMEI. Deleting web cookies, the other component of our privacy control, is straightforward to implement using existing WebKit hooks.

To find the names of the packages to restrict, we analyzed the `.jar` files of popular advertising and analytics libraries. We used these package names to configure a restricted list; since the popular trackers are used by many applications a short list gives good coverage.

5.2.2 Evaluation

In order to estimate effectiveness of the selective data shadowing mechanism, we first analyze which packages in applications access privacy-sensitive resources. If only third-party packages access those resources in an application, we would not need the selective data shadowing (i.e., we can safely use monolithic data shadowing per application).

Table 6: The number of applications accessing IMEI, AndroidId and Location.

	IMEI	AndroidId	Location
accessing resource	319	526	643
via third-party packages	109	440	482
via a **mixed set**†	38 (12%)	160 (30%)	155 (24%)

†a mixed set of primary and third-party packages

We use the Android `apktool`[4] to disassemble and inspect 1100 popular Android applications[5]. Through manual inspection of disassembled codes, we found 12 adver-

[4]http://code.google.com/p/android-apktool/

[5]We obtained the 50 most popular applications from 22 categories in Nov. 2010. It is the same set of applications with those used in [13].

tising and analytics packages: com.admob, com.flurry, com.google.ads, com.mobclix, com.mobclick, com.qwapi, com.smaato, com.inmobi, com.nexage, com.jumptap, com.tapjoy, com.millennialmedia. Table 6 shows the number of applications accessing IMEI, AndroidId and Location. Each row represents # of apps that access the resource; # of apps that access it with at least one third-party package; # of apps that access it via a mixed set of third-party and primary packages. From the table, applications in the last row accessing the resources via a mixed set requires the selective data shadowing if users would like to opt-out of third-party tracking. They comprise 12%, 30% and 24% of applications accessing IMEI, AndroidId and Location, respectively, which shows that a non-negligible number of applications can benefit from our privacy control mechanism.

For the evaluation of the selective data shadowing, we look into whether the mechanism is able to safely prevent application from leaking personal data to third-party servers and whether it causes any side effects. As observed in [13], enforcing a new privacy control may introduce side effects in some applications. It is because of the sloppy programming practice in which applications are not equipped to cope with the unavailability of the requested data.

We use 30 applications and TEMA[6] scripts written for them in the evaluation of AppFence [13]. The scripts automate the execution of these apps and generate screen captures so that we can characterize visible side effects. In addition, using the scripts allows us to directly compare the result with AppFence's as the execution paths are kept unchanged. To ensure correctness, we add a new set of taint tags for shadowed data, so that we can check that only shadowed data is sent to third-party services while original data is sent to other services as AppLog records all the communication with taint tags.

We configure our privacy control to shadow IMEI, AndroidId and Location if they are accessed by the predefined third-party packages which we listed above. After running all 30 applications, we inspect screenshots and AppLog records, and compare the results to the results of AppFence's two configurations: monolithic data shadowing, exfiltration blocking to known advertising and analytics (A&A) servers. AppFence's exfiltration blocking mechanism drops messages which contain tainted data, and the configuration of exfiltration blocking to known A&A servers was shown to have the least amount of side effects in its evaluation—advertisement disappeared for a few applications.

Table 7 presents the number of applications having side effects caused by the three different configurations. Note that the number of applications having ads absent with exfiltration blocking has increased because AppFence did not taint AndroidId. As shown in the table, the selective data shadowing caused no side effects for the all 30 applications. By contrast, 12 (40%) applications with monolithic data shadowing suffered from side effects. The side effects come from that applications cannot access current location, persistent IDs or contacts and the applications require the information for their functionality. 17 (56%) applications with exfiltration blocking lost their advertisements as many advertising messages contain either of IMEI or AndroidId. Also, the selective data shadowing mechanism successfully shadowed data to third-party servers in all cases while allowing applications to send original data to other servers. Note that 5 of 30 applications try to send messages containing same resource to third-party servers and others.

[6]http://tema.cs.tut.fi/intro.html

Table 7: The side effects of imposing three different privacy controls. S.Shadow: Selective data shadowing, AppFence1: Monolithic data shadowing, AppFence2: exfiltration blocking to known A&A servers

	S.Shadow	AppFence1	AppFence2
None	30	18	13
Ads absent	0	0	17
Other Effects	0	12	0

Since it relies on app structuring conventions there is no guarantee that our control will work for all apps. For example, an app might store tracking identifiers like IMEI in its own database that we cannot reset, or a tracking library might access location directly from the rest of the app instead of via the Android API. Nonetheless, we believe the mechanism is likely to work well for a large number of apps.

The kind of opt-out that it provides depends on how the restricted packages are shadowed. For example, if per-app salted IMEI is returned instead of the true IMEI then the third-party may still track the user, but loses the ability to track across applications. If instead a random IMEI is returned on each run then the third-party loses the ability to track the user across different sessions. Similarly, tracking is limited to a single session when web cookies are deleted each time the app is started. Interestingly, the privacy of this opt-out is stronger than blocking third party cookies on the web: even when third-party cookies are blocked, third parties may track a user across different sessions on the same site.

5.2.3 *Future directions*

We consider the initial prototype to be promising, and plan to develop it further. One direction is to make the privacy control easily accessible to non-technically expert consumers, since it is specifically intended to support their needs. For this, we plan to iterate a UI design through usability studies. Since our approach relies on a list of known third-party advertising and analytics components, there is also the question of how to keep this list up to date. Static analysis of applications using tools such as ded [9] will surely help, and we are investigating automatic ways to identify third-party code within an application binary. Finally, it would be useful to broaden the boundary of this privacy control to make it more difficult for third-party companies to defeat it by colluding with application developers (e.g., by creating a new API to pass tracking identifiers from app code directly to advertising code). It is likely that static call graph analysis (possibly with binary re-writing) and dynamic taint checking can find and enforce stronger boundaries between application components. We believe this is an interesting direction for research to help users prevent unwanted third-party tracking.

6. RELATED WORK

Prior work that investigated smartphone apps has highlighted privacy issues with the exposure of personal information and persistent identifiers. This work ranges from informal investigations by the Wall Street Journal [4] to the use of static analysis techniques to find leaks that may occur in usage [7, 9]. Our study differs by measuring the tracking that is part of everyday mobile usage outside of the lab, and by taking a comprehensive look across persistent identifiers and traditional tracking by web cookies.

Tracking has been studied in the complementary context of the web [14, 15, 20]. This setting is relatively well understood compared to mobile tracking, and it has helped to inform our research. Tracking defenses also exist in the web context, from browser cookie mechanisms and add-ons to DoNotTrack [16] and ShareMeNot [20]. These defenses do not apply to mobile tracking, which led us to develop new privacy controls.

Our study relies heavily on dynamic information flow tracking to measure whether personal information and persistent identifiers are exposed by apps. In particular, we extend TaintDroid system [8]. Taint techniques have also been used for privacy defenses, such as our work on blocking the exposure of sensitive information [13]. However, the new privacy control we explore does not require tainting. Another privacy defense, MockDroid [6] provides a way to give applications shadow resources including device ID, but it does not differentiate third parties. Our selective data shadowing takes a similar approach with the extended stack introspection of Java virtual machines in the early web browsers [22]. When a dangerous resource is accessed, it checks each stack frame to ensure the access is made through previleged code.

Other research is developing alternative designs for advertising that are compatible with privacy, such as Privad [12] and auctions that are compliant with Do-Not-Track [19]. These designs are not yet targeted at mobiles.

Finally, other mobile phone studies are improving our understanding of everyday mobile phone usage. These studies now have large user populations (e.g., 255 users in [10]) or observe behavior over long periods of time (e.g., LiveLab [18]). None of these studies focus on mobile tracking.

7. CONCLUSION

We have presented the first field study of how real-world users are being tracked by the apps that they run on their Android smartphones. We instrumented mobiles with dynamic information flow tracking to collect records of when sensitive information was sent off the device. We then recruited 20 participants to use these mobiles as their primary phone for everyday tasks for a period of three weeks in November and December 2011.

We see extensive mobile tracking in the data that we collected. Tracking is embedded into the app by the developer in 75% (168 out of 223) of the apps that participants used during the study. Advertising and analytics are common as third-party trackers. They were embedded in 57% (127 out of 223) of the apps so that every participant in our study was tracked multiple times. Perhaps more concerning is the way tracking is often implemented using persistent identifiers that can be linked across applications and tracking companies. As well as the previously described use of IMEI for tracking, we found the use AndroidID—a long-lived device identifier that requires no user permission to access—to be common. It was sent to 74 domains versus 52 for IMEI. The most prevalent 10 advertisers in our study often sent these identifiers as literals along with the mobile's current location and without the protection of encryption. The privacy risks of this kind of mobile tracking seem incongruous in a regulatory environment in which consumers are gaining the ability to opt out of web tracking with Do Not Track.

On a more positive note, the interviews that were part of our study helped us to understand the kind of privacy controls that consumers want. While the participants expected to be tracked, they wanted greater transparency to know when they were tracked, and the ability to opt out when they did not perceive value. Unfortunately, the existing Android OS does not give users any way to control third-party tracking in an app separately from the application itself. We prototyped a new privacy control to let users exercise this control. We find it promising enough in initial tests that we plan to develop it further to help users in the mobile tracking arms' race.

Acknowledgements

We would like to thank Microsoft Research for providing equipment, and Daniel Halperin, Vincent Liu, Xiao Wang and Will Scott for providing valuable comments. This work was supported by National Science Foundation award CNS-0917341.

8. REFERENCES

[1] Best practices for handling android user data. `http://android-developers.blogspot.com/2010/08/best-practices-for-handling-android.html`, 2010.

[2] Opt out of behavioral advertising. `http://www.networkadvertising.org/managing/opt_out.asp`, 2011.

[3] Same origin policy. `http://www.w3c.org/Security/wiki/Same_Origin_Policy`, 2011.

[4] S. T. Amir Efrati and D. Searcey. Mobile-app makers face U.S. privacy investigation. `http://online.wsj.com/article/SB10001424052748703806304576242923804770968.html`, Apr. 5, 2011.

[5] A. Barth. RFC 6265: HTTP state management mechanism. `http://tools.ietf.org/html/rfc6265`, 2011.

[6] A. R. Beresford, A. Rice, N. Skehin, and R. Sohan. MockDroid: Trading privacy for application functionality on smartphones. In *Proceedings of the 12th Workshop on Mobile Computing Systems and Applications (HotMobile)*, 2011.

[7] M. Egele, C. Kruegel, E. Kirda, and G. Vigna. PiOS: Detecting privacy leaks in iOS applications. In *NDSS*, 2011.

[8] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. TaintDroid: An information-flow tracking system for realtime privacy monitoring on smartphones. In *OSDI*, 2010.

[9] W. Enck, D. Octeau, P. McDaniel, and S. Chaudhuri. A study of android application security. In *Usenix Security*, 2011.

[10] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin. Diversity in smartphone usage. In *Mobisys*, 2010.

[11] R. Fielding. Tracking preference expression (dnt). W3C Working Draft, 2011.

[12] S. Guha, B. Cheng, and P. Francis. Privad: Practical privacy in online advertising. In *NSDI*, 2011.

[13] P. Hornyack, S. Han, J. Jung, S. Schechter, and D. Wetherall. "These aren't the droids you're looking for": Retrofitting android to protect data from imperious applications. In *CCS*, 2011.

[14] B. Krishnamurthy and C. E. Willis. Generating a privacy footprint on the internet. In *IMC*, 2006.

[15] B. Krishnamurthy and C. E. Willis. Privacy diffusion on the web: a longitudinal perspective. In *WWW*, 2009.

[16] J. Mayer and A. Narayanan. Do not track-universal web tracking opt out. `http://donottrack.us/`.

[17] A. M. McDonald and L. F. Cranor. Americans' Attitudes About Internet Behavioral Advertising Practices. In *WPES*, 2010.

[18] A. Rahmati, C. Shepard, C. Tossell, M. Dong, Z. Wang, L. Zhong, and P. Kortum. Tales of 34 iphone users: How they change and why they are different. Technical Report TR-2011-062, Rice University, 2011.

[19] A. Reznichenko, S. Guha, and P. Francis. Auctions in do-not-track compliant internet advertising. In *CCS*, 2011.

[20] F. Roesner, T. Kohno, and D. Wetherall. Detecting and Defending Against Third-Party Tracking on the Web. In *NSDI*, 2012.

[21] S. Thurm and Y. I. Kane. Your apps are watching you. The Wall Street Journal. Dec. 18, 2010. `http://online.wsj.com/article/SB10001424052748704694004576020083703574602.html`.

[22] D. S. Wallach, D. Balfanz, D. Dean, and E. W. Felten. Extensible security architectures for java. In *Proceedings of the sixteenth ACM symposium on Operating systems principles*, SOSP '97, pages 116–128, New York, NY, USA, 1997. ACM.