Re-identification of Mobile Devices Using Real-Time Bidding Advertising Networks

Keen Sung, JianYi Huang, Mark D. Corner, Brian N. Levine
University of Massachusetts Amherst

ABSTRACT

Advertisers gather data about users and their mobile devices through ads placed within Android and iOS apps. Most of the time, location, device, and app information are linked to the same device using a unique advertising ID (Ad ID). If the Ad ID is not available, advertisers can still use geo-coordinates or IP address to infer links in data gathered from different ad placements.

Even though the Ad ID can be disabled by users on both OSes, we demonstrate that advertisers can leave their own unique strings (*marks*) in the app storage, and use these strings to link information collected from ads. Users cannot clear these marks without losing all data within the app. Because advertising platforms allow connection filtering and geofencing, users who either connect using a non-cellular IP address or allow location access to the app are substantially easier to be rediscovered by the advertiser.

We carried out many large-scale experiments on iOS and Android devices involving hundreds of thousands of impressions. We found that on average 49% of impressions from an iOS device, and 59% of Android impressions could be re-identified for less than $5/day per device using this strategy. We subsequently verified this method on 1,727 devices and recovered 660 of them within 48 hours for $86.73. Finally, we explore the behavior of privacy-seeking VPN users. We found that for the majority, their clearnet IP address and location could be unmasked easily using ads.

CCS CONCEPTS

• **Security and privacy → Mobile and wireless security**; **Software security engineering**.

KEYWORDS

Mobile Advertising; VPN; Cellular; Security; Privacy

ACM Reference Format:
Keen Sung, JianYi Huang, Mark D. Corner, Brian N. Levine. 2020. Re-identification of Mobile Devices Using Real-Time Bidding Advertising Networks. In *MobiCom 2020 (MobiCom '20), September 21–25, 2020, London, United Kingdom.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3372224.3419205

MobiCom '20, September 21–25, 2020, London, United Kingdom
© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-7085-1/20/09…$15.00
https://doi.org/10.1145/3372224.3419205

1 INTRODUCTION

By using mobile apps supported by advertisements, users allow fine-grained information to be shared with hundreds of entities in the advertising ecosystem. Advertising libraries embedded in apps facilitate the dissemination of the device's unique *mobile advertising identifier* (Ad ID), IP address, user-agent string, application name, and fine-grained GPS coordinates to entities that bid for ad space via open *Real-Time Bidding* (RTB) marketplaces [31].

However, as we show in this paper, advertisers can still *re-identify* devices in RTB networks. The lack of controls in web storage APIs allow advertisers to leave a retrievable *mark* on the user's device, retarget groups of devices that may contain the user, and then verify the mark when an ad is shown. This re-identification of users can happen even when the user disables or changes the Ad ID, uses a VPN, or disallows location.

In this paper, we quantify the cost and effectiveness of the tools available to RTB advertisers to re-identify mobile devices carried by users, including cases where the user has taken steps to prevent it. We show that to fully control exposure of identifying information, a user must cobble together several techniques, and they must be used flawlessly. Accordingly, the likelihood that an advertiser can re-identify devices over time is very high and typically inexpensive.

- First, a user can be uniquely re-identified by the Ad ID. To control this technique, the user must opt out of Ad ID tracking on iOS and Android. In iOS 14, this setting will be on by default on the app level, and users will be explicitly asked to opt in to Ad ID tracking within an app before advertisers are able to do so [4]. Enabling iOS's *Limit Ad Tracking* (LAT) setting zeros out the identifier, making the device indistinguishable from other users of that feature [7]. On Android, enabling the *opt out of ad personalization* (OOAP) feature is merely an advisory flag that apps and advertisers are expected by Google to respect [1]. It's a limited tool for users, and continuously resetting the Ad ID via a well-buried button is simply impractical.
- Second, advertisers can make use of various methods of *marking* users (e.g., LocalStorage, IndexedDB, or cookies). The mark can be used by advertiser to link observations of the same device despite a reset or zeroed-out Ad ID. There are no user-facing methods to control marks and to a large extent Android and iOS do not allow applications to prevent advertisers from using those APIs.
- Third, advertisers can leverage features to re-target devices, including device model, app, and the IP address. Further, some IP addresses can be coarsely geolocated to approximate locations, and many advertising services allow ads to be targeted within geographic fences. Additionally, advertisers can make use of precise GPS coordinates, if the user consents to it, and many do.

Users can only affect this strategy if they use a VPN all of the time and disallow location access in the app.

Contributions. We evaluate the cost and effectiveness of these re-identification strategies by deploying hundreds of thousands of ad impressions on a self-service RTB platform. An impression is a single appearance of an advertising creative, such as a small banner or a full screen "interstitial". Our focus is an advertiser that seeks to maximize the number of impressions attributable to a specific mobile device, even when the user is trying to prevent re-identification. Our specific contributions, based on a mixture of empirical results from real campaigns and also *post hoc* analysis, are as follows.

- We detail app-based techniques for re-identification. To our knowledge, we are the first to discover and report that the in-app web views that display ads can be used to store marks. It is unclear how to disable the storage, or how to clear the marks other than deleting and reinstalling the app. We quantify the lifetime of seven methods of storing marks in apps. Using one marking technique at a time, 70% of marks persisted at least one day, and 35% of marks persisted at least 12 weeks. Combining the techniques increases the rate of successful marking.
- We quantify the efficacy of *disabling Ad IDs* on devices (some have suggested resetting it monthly [28], and the Ad ID is disabled by default in iOS 14 apps) given that marks can be used to link observations of the same device. Without the Ad ID, 75% of Android devices can be re-identified in the first week using marks, and 81% of iOS devices. Even 30 days later, 61% of our Android impressions and 71% of iOS contained marks to re-identify the device.
- We show the precision of re-identification based on *retargeting ads* to three types of features: IP addresses; geographic locations; and the static features of app name, model and OS. Devices were found again on previously visited non-cellular IPs 78% of the time. Devices remained within a 10 km radius 80% of the time over two weeks. Devices with iOS are significantly more difficult to re-identify because Apple device models are indistinguishable during retargeting.
- We perform two case studies. First, we determined the cost and performance of attempting to rediscover over a thousand devices using information from only one impression, and without using the Ad ID at all. Second, we show that ads can very effectively reveal the clearnet IPs (and locations) of users of BitTorrent apps masked by a VPN service.

We begin by providing an overview of mobile advertising and defining the adversarial model that the advertiser follows in our experiments.

2 BACKGROUND

There are many modalities of purchasing advertisements, from walled-garden advertising systems such as Facebook, to relatively open *real-time bidding* (RTB) networks. In RTB, apps that want to sell ad space (i.e., *publishers*) work with *Supply Side Platforms* (SSPs), and entities that wish to show their ads (i.e., advertisers) work with *Demand Side Platforms* (DSPs). When an app shows an advertisement, it contacts one or more SSPs. The SSP brokers an auction amongst a set of DSPs who bid on behalf of advertisers. The highest bidder wins, and the winning ad (known as a *creative*) is sent to the user's device. When the app shows the creative, it is called an *impression*. This auction happens in fractions of a second and is summarized in Figure 1. Further details of the RTB ecosystem can be found in works by Vines et al. [36] and Corner et al. [20, 21]. Small ads are very inexpensive, e.g., typically \$0.10–\$5.00 per 1,000 shown (*cost per mille* or CPM, in industry parlance).

Information return. Once a bid has been won, the device is sent a snippet of HTML code, commonly called an *ad tag*, and the app displays it within an embedded web view. This snippet is a small HTML document that the device loads; a sample of our deployed ad tag is shown in Appendix A. This HTML contains an image tag for the advertiser's creative image, but can also contain any other element found in typical webpages, including invisible 1x1 images, commonly called "tracking pixels". The webview fetches these pixels directly from the advertisers' servers allowing the advertiser to verify that the impression was successful.

Additionally, in the URL for the pixel, the advertiser can include *macros*. Before the DSP passes the advertiser's HTML ad tag to the SSP (who passes it to the device), the DSP substitutes these macros with values from the original ad bid request. This allows the advertiser to learn fine-grained details of the device the ad appeared on, including Ad ID (when available), the time, the service provider, location (when available), the app, and some device information. Ad IDs are 16-byte UUIDs and are called the *Identifier for Advertisers* (IDFA) in iOS and the *Google Ad ID* in Android. The location information is provided by the device GPS when it is available or an IP-based geolocation service.

By including and serving an image in the tag, the advertiser can collect the device's IP address. Device information can be input as query string parameters in the image's URL, and the advertiser keeps a log of resource requests that would include the URL and the connecting IP address. If the advertiser returns JavaScript as part of the ad, then it can access browser APIs, such as storage in IndexedDB and LocalStorage.

An advertiser can collect information and access browser APIs any time an advertisement is shown. While this only happens while the app is running (advertisements aren't loaded while the app is in the background), it does not require the user to click on the ad, the ad just has to be loaded. Additionally, there is no indication to the user what information the advertisement collects and disseminates separate from the app itself. An application that collects precise location information may have asked the user for permission long before an advertisement collects it.

Device retargeting. Advertisers on RTB systems can target and retarget devices based on features such as location, IP address, ISP, app, device, and time of day. Advertisers often use Ad IDs to retarget the same device over time. Ad IDs are unique to the device and shared across all apps on the device (and not available from ads viewed with a normal web browser). These device-level identifiers are very powerful as they are consistent across all apps on the device. Users have some control over their privacy: on both Android and iOS, users can reset the ID at any time, and on iOS the Ad ID can be disabled outright.

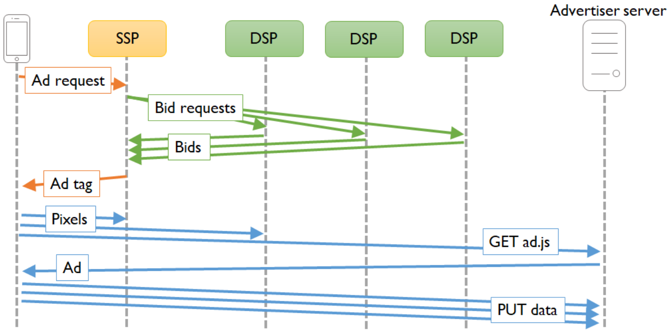

Figure 1: Sequence diagram of a single real-time bidding cycle

Devices can also be retargeted by the coarse geolocation or their precise GPS location. Weather and dating apps often share precise GPS [21], and surprisingly so do BitTorrent apps. Home and work IP addresses typically match a small number of devices and are often very persistent [33]. But even in the case of shared IP addresses, such as those at a library or coffee shop, the geolocation of the IP address massively narrows down the number of possible users while retargeting devices. Cellular IPs are much broader as geolocating such IPs could be several states wide in the USA [17]. We discuss location in more detail in Section 4.2.

3 ADVERSARIAL MODEL

Our adversary uses RTB to place advertising impressions on mobile devices with the following goal:

Goal: *to maximize the recall of impressions on a particular mobile device even when the device's user tries to prevent re-identification (such as resetting or clearing the Ad ID, denying the application access to location, or using a VPN).*

Our goal is not to optimize cost, but we favor solutions that are less expensive.

The user may reset their Ad ID periodically or disable it (setting it to all zeros). The user may make use of many IP addresses as a natural consequence of movement, or because of their use of a cellular NAT, or a VPN. The user may explicitly allow access to GPS locations, but cannot prevent the advertiser from using a geo-location service based on their IP address.

We evaluate an adversary's success rate and cost of re-identifying a device over the course of a month. During that time, the user can either disable her Ad ID, reset it daily, or reset it weekly. The adversary does not aim to discover the true identity of the user; rather, the adversary tries to associate anonymous behaviour with previous activity.

The most obvious defense against the advertiser (excluding buying the ad-free version of an app) is to use a VPN, which can mask the user's *clearnet* IP address (i.e., an address assigned by an ISP that maintains DHCP records and accurate billing information). Thus, in Section 6.2, we explore an alternate goal: to determine the clearnet IP address (and therefore geographic locale) of a device masked by a VPN.

Self-service vantage point. In RTB systems, there are multiple vantage points to observe users: as the developer of the advertising SDK in the app, the SSP, the DSP, and as a "self-service" user of a DSP. The developer of the advertising SDK can trivially track users as they have full native access to the device. The SSP and DSP see all bid requests coming from the device and can track users at no additional economic cost, though they can only access data available in the bid request. We have chosen to analyze the *weakest* vantage point: that of the self-service user. One can sign up with a self-service DSP easily; minimum spend is typically $100USD. A self-service adversary can target ads to users, but only gets feedback when an impression is won. Therefore, collecting data has an economic cost, typically 10 cents to $5 for each 1,000 ads (called a cost per mille, or CPM).

As we show in Sections 6.1 and 6.2, reidentifying users can be done for pennies per user. Knowing if this economic cost is reasonable depends on the adversary's motives. While seeming inexpensive, it is more than an advertiser would typically pay to retarget advertisements. However, this cost is inexpensive for a more motivated adversary seeking to investigate a criminal perpetrator, or trying to re-identify devices for targeted fraud, state-sponsored espionage, and cyber-stalking.

4 METHODOLOGY

To evaluate the ability of advertisers to meet the adversarial goals we define in Section 3, we implemented a strategy for an adversary to re-identify users in RTB networks. Starting from an initial ad impression, the adversary uniquely *marks* the device, then later retargets advertisements to a profile that includes the original device, and finally confirms which device was the original one by reading the mark. Here, we detail our implementation for marking devices and then retargeting them to rediscover particular devices.

4.1 Marking implementation

We have discovered that the same APIs available to browser-based pages are available unheeded to advertisements. Because in-app advertisements are shown within embedded web views, ads can use any browser-based method of storing unique identifiers in the device and using it as an identification mark.

We identified and implemented seven different methods of marking devices, listed in Figure 2. In our implementation, we had access to JavaScript-based ads, giving us access to browser APIs for the first four features: LocalStorage, IndexedDB, Web SQL, and the cache. As a fifth feature, our implementation served unique JavaScript files to each user that contained the mark; that mark is accessible when the cached JavaScript is executed in subsequent ads.

In some cases, an advertiser may not have access to serving JavaScript as part of the advertisement. In those cases the advertiser typically uses 1x1 tracking pixels, which the advertisement fetches from the advertiser's servers. To support these cases, we implemented our sixth and seventh marking methods, respectively: set and retrieve a cookie when responding to the tracking pixel; and HTTP ETags, which browsers transmit to the server to re-validate an expired cache item. The ETag is set in one ad, and then sent to the server by the phone in a subsequent ad.

Feature	**JS**	**Non-JS**	**iOS**	**Android**
1. LocalStorage	✓		✓	
2. IndexedDB	✓		✓	✓
3. Web SQL	✓		✓∗	✓
4. Cache API	✓		✓∗	✓∗
5. Unique, Cached JS file	✓		✓	✓
6. Cookies	✓	✓	✓∗	✓
7. HTTP ETag	✓	✓	✓	✓

Figure 2: Seven storage-based marking features that we evaluated. ✓∗ indicates partial support.

If either the user or developer of the ad SDK can prevent or erase such marks, then user privacy would be increased: the adversary cannot know that they found the exact device again, only that it is one of the many devices in the re-targeted group.

For a user to prevent marking, they have three coarse options: (1) they can clear stored information by deleting all of the app's data and reinstalling; (2) they can buy a version of the app that has no advertisements, if available; or (3) not use the app at all.

A better option for users would be if developers of ad SDKs configured the web view to enable or disable the features in Figure 2. (As we show in Section 5, at least one storage method is typically available in ad SDK web views.) Completely disabling web view storage is non-trivial for developers on both iOS and Android. The programming calls to disable are not well documented; we discuss a method we discovered through testing in Appendix B.

A disadvantage of re-indentifying devices with web view-based marks is that each app is sandboxed. Tags stored on one of these in-app web views cannot be seen by other apps. This means that all re-identification of devices must take place within a single app.

4.2 Retargeting strategy

The advertiser has to balance re-identifying the user with the economic cost. A typical DSP gives access to billions of advertising opportunities per day. When the Ad ID is not available, the advertiser must avoid advertising to such a large crowd that it becomes economically infeasible.

Figure 3 lists the retargeting features available to us from the DSP we used. We filter impressions so that a minimum number of impressions is required to re-identify a single user. For example, IP addresses in homes (e.g., via cable) typically do not change over time; and it is easy to retarget a device from that home by the address. Such a strategy will, however, miss opportunities to retarget the user's device at their workplace. Targeting a radius of geographic locations around a first sighting of a device captures more possibilities than a home IP alone, although at an increased cost. Costs can be reduced by refining the advertising targeting by including other factors, such as the device operating system, the bundle identifier for the application they use, and the device model.

Somewhat ironically, when users enable Limit Ad-Tracking on their devices, it can make them more identifiable — only one out of six users enable that option [8]. The particular DSP we used did not allow us to target an empty string or zero ID. If other DSPs allow that, one could use the zero identifier as a way to filter users. The impact of iOS 14's change to make Ad ID opt-in, rather than opt-out, could not be quantified at the time of this paper.

Feature	**Prop. matched**
Device OS	0.51
Bundle	0.24
OS Version	0.18
Android Model	0.012
GPS	1.1×10^{-4}
IP	6.3×10^{-5}

Figure 3: Retargeting features available to advertisers, and the average proportion of devices matching a feature. More specific filters result in cheaper campaigns. For example, targeting a specific device OS can halve the cost of a campaign. GPS and IP address are the most cost effective retargeting features, and the cost is further reduced by combining all of these features. Further discussion is in Section 5.3.

5 EVALUATION

Our evaluation is focused on quantifying the advertiser's success rate and economic cost of re-identification. There are two key metrics that we use throughout the evaluation: impression rate and recall. The impression rate is the number of extra impressions allowed by the retargeting filter when trying to rediscover a device. For instance if we use a filter that targets iOS devices using a particular IP address, any auctions we win for iOS devices on that IP other than the device are overhead.

Recall measures our ability to find a device again. There are several reasons why we might not find a device: (1) the device matches the filter but we are unable to reidentify them as the same device because the mark is lost; (2) the device isn't used again during the study period; (3) it is used again, but does not match the retargeting filter, such as using another IP address; or (4) we lose the ad impression auction for some reason.

We conducted the evaluation through a self-service DSP using banner advertisements purchased in a wide variety of mobile applications. We collected several data sets, comprising hundreds of thousands of ad impressions using a variety of targeting techniques. We begin with a description of the data we collected, and our evaluation comprises three results:

- First, we determined the *persistence* of each marking method; we looked at how long a mark would remain on a phone without renewing it.
- Second, we determined how often users could be rediscovered (i.e. recalled) over one month, compared to Ad ID targeting. We also performed *post hoc* analyses to evaluate the relationship between cost and recall.

We also conducted two case studies, in Section 6.2. One of these is an end-to-end measure of cost and efficacy in attempting to rediscover 1,727 users without using Ad ID, from only one impression. Another is an evaluation of the behaviour and privacy of VPN users.

Data set	Days	Imps	Ad IDs	Filter	Section
PERSIST	81	52,719	1,715	Ad ID	§5.2
LONG	33	108,084	1,384	Ad ID	§5.2
IP	3	72,508	44,055	IP	§5.3
GPS	3	44,570	28,673	GPS	§5.3
FIRST	4	22,319	9,986	IP/GPS	§6.1
VPN	28	74,026	13,778	Ad ID, App	§6.2

Figure 4: Each data set targeted a specific hypothesis. "Filter" indicates the type of parameter used for retargeting.

5.1 Data sets

We created an account on a self-service DSP and purchased JavaScript-based banner advertisements in hundreds of Android and iOS applications. These included games, weather applications, chat, and dating apps. Embedded in the advertisement's HTML code were seven different marking methods, as well as impression macros to return the device's IP address, geolocation, user-agent string, OS version, and any available Ad ID. While the end goal of our advertising adversary is to re-identify devices even if users rotate or block their Ad ID, we use the Ad ID as the ground truth for our experiments. This ground truth allowed us to know with certainty that we re-identifed the user or if we failed to re-identify them via retargeting. Our bids were a maximum of $10 CPM; the average winning bid was $4.02025 CPM. We bid extremely high to increase our data set.

The data sets we collected are summarized in Figure 4. We targeted ads to ten United States municipalities over a period of two days. To avoid bias due to geography or population density, we randomly selected eight micropolitan statistical areas [10], and added New York and Los Angeles[1]. From these two days, we randomly sampled a set of 3,099 Ad IDs and split the set into two data sets.

(1) We placed 1,715 devices into our PERSIST data set. We retargeted each of these using their Ad ID after 7, 20, 70, and 81 days from the original impression, with no impression in between. The purpose of this data set was to verify the persistence of different marking methods.
(2) We placed the remaining 1,384 devices in our *LONG* data set. These were retargeted every two hours over a period of one month. We targeted devices that did not disable the Ad ID. We used the devices' features (such as IP address) to retarget ads and compare the performance of our strategy to Ad ID retargeting.

The overhead cost of this rediscovery strategy is proportional to the impression rate. We deployed several campaigns to measure the impression rates (the number of impressions per time period we won on devices other than the target device) of various advertising filters. We used these measurements in conjunction with the LONG data set to analyze *post hoc* the cost and efficacy of several scenarios.

- Our IP data set consists of impressions collected from targeting a sample of 920 IP addresses from devices in the LONG data set. Addresses were collected from the first five days (i.e. the training period) and were used to quantify the cost of retargeting by IP.
- Our GPS data set consists of impressions from retargeting a 10 km radius within eight towns and two cities. From this data, we determined the correlation between impression rate and population density.

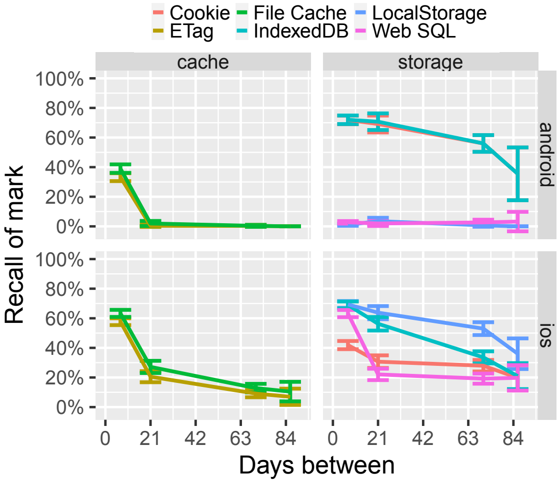

Figure 5: Proportion of marks found on devices after not seeing these devices in the intervening period.

We then conduct two case studies.

(1) In our FIRST data set, we attempted to rediscover 1,727 devices from five US towns using information from only the first impression, and without using Ad ID at all. While 1,041 devices of these devices reported Ad ID, which we ignored, 686 did not report one at all.
(2) Finally, we collected data from devices using a VPN, discussed in Section 6.2.

It is important to note that all of our data is taken from a single RTB DSP at a single point in time. Although this DSP aggregates a large number of SSPs, a lot of data remains hidden from us, including so-called "private auctions" [9], other SSPs not aggregated by this DSP, or because publishers block our ads due to their content (higher education). While our results are likely typical, it is hard to measure a global, market-driven system of 7 billion devices in a fully repeatable way. The advertising market also changes rapidly due to seasonality [13], the recent COVID-19 pandemic [15], and other short- and long-term trends.

5.2 Mark persistence

The persistence of marks varies between storage mechanisms, and the adversary's goal is to re-mark a device before the original mark is cleared. For example, the mark may be cleared when the app has exceeded its assigned quota [5], or will be cleared when an app or OS is reinstalled or updated.

To measure how long marks persist, we used our PERSIST data set. We set marks using the seven techniques (described in Section 4.1) and intermittently retargeted the same set of Ad IDs to check if the marks disappeared. Figure 5.2 shows these results. While caching marks were mostly cleared after one week, storage

[1] The other locations were Kearney, NE; Fitzgerald, GA; Cadillac, MI; Beeville, TX; Winnemucca, NV; Platteville, WI; Durango, CO; and Alexandria, MN.

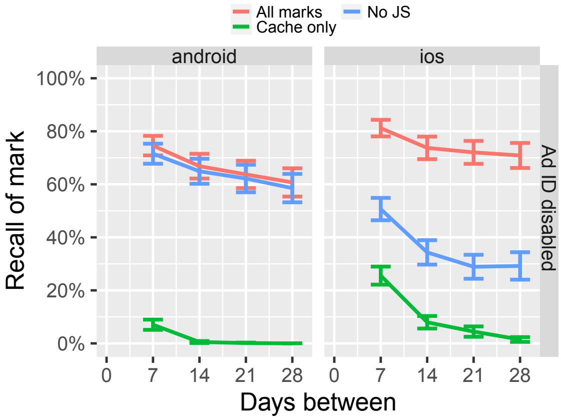

Figure 6: Proportion of marks found on devices while retargeting them up to twelve times a day, allowing marks to be stored anew each time.

methods lasted much longer. On iOS, LocalStorage was most usable, seen again in 69.3% of impressions after a week. On Android, LocalStorage is disabled by default, but IndexedDB and cookies were both available on most apps, seen 72.0% of the time. More than half of the most persistent marks were still retrievable after two months. We used the LONG data set to show that marks are more persistent if devices are encountered regularly. As shown in Figure 6, the non-JS method, HTTP cookies, is more persistent on Android when compared with iOS.

Unless all stored marks are cleared at the same time, the missing values can be restored using the existing identifiers, similar to the *evercookie* [26]. Using the LONG data set, we quantified the persistence of overall marking if users were retargeted every two hours for a month. These results are summarized in Figure 6. All marks on Android devices had about the same persistence as IndexedDB or cookies alone, indicating that when marks are cleared, they are all cleared at once. On the other hand, iOS impressions were recalled at a significantly higher rate, remaining at 75% recall after one month, indicating that different marks were cleared at different times.

5.3 Retargeting

The cost to recall a device depends on the retargeting filter; more permissive filters are more likely to find a particular device again, but also increase the number of non-target devices and consequently the overall cost. IP targeting can be very effective if a device connects using a consistent IP address. Geocoordinate targeting is also useful since users generally stay in the same area.

We investigated a strategy of targeting the IP addresses seen during training, additionally targeting some radius of the GPS coordinates seen, and filtering out irrelevant bids based on static features (device model, OS, and app used). The success of this strategy depends on the devices' impressions matching consistent geographic coordinates and IP addresses over time, as well as consistent use of apps.

Our basis for evaluating cost was impression rate (i.e., number of extraneous impressions per day) measured over three days of testing. In both GPS and IP data sets, we attempted to limit impressions to one per device using a flag provided by the DSP. This impression limiting does not work perfectly; devices that do not share an Ad ID could be counted more than once. We expected the impression rate to go down, but running a longitudinal experiment to quantify this decrease was not financially viable.

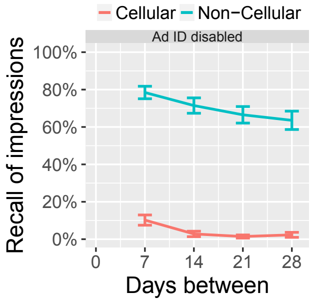

Figure 7: Device recall using IP, given different resetting policies. The dotted lines are cellular IPs, solid line is non-cellular.

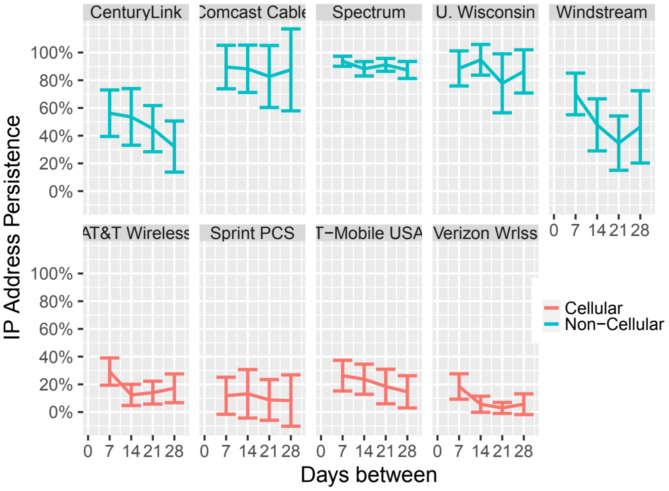

Figure 8: Persistence of IP address from various ISP.

IP addresses. We first evaluated the persistence and specificity of IP addresses used by mobile devices. We expected that many non-cellular IP addresses would lead to a higher recall, and incur a low overhead of unwanted impressions, and we expected the opposite to be true of cellular IP addresses. The latter tends to assign temporary IPs from a pool of addresses with many users sharing a particular IP.

We considered the first five days of the LONG data set to be the training period, and evaluated the overall usage of those IPs by each device during the next four weeks. Using MaxMind [12], we determined the ISP of each IP, and labelled them as cellular or non-cellular. In Figure 7, we show that non-cellular IP is a very effective retargeting feature: in many cases, users return to *sticky IPs* [33], addresses that are assigned to a modem long term. These IPs have a much higher recall than cellular IPs which are typically

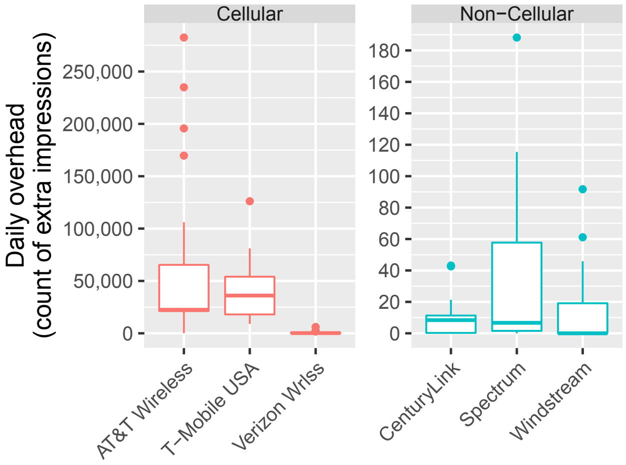

Figure 9: Extra impressions per day for an IP, with Ad ID capping. Values shows the median. Many of our Windstream IPs exhibited no extra impressions.

ephemeral. Figure 8 shows a breakdown of the same IP recall by ISP, which further highlights the amount of privacy gained simply by using a cellular ISP. Non-cellular IPs from some ISPs are found 80–90% of the time over the entire month, while cellular IPs are rarely useful with less than 10% recall.

However, we must also consider how many other devices may be using the same IP address to determine how expensive it will be to retarget a particular device. Figure 9 shows the impression rate of different ISPs, as determined from our IP data. The impression rate is the number of impressions found using a particular advertising filter, in this case a single IP address. Note that this isn't the number of unique devices found at that IP, but rather how many extra impressions we will have to buy using that filter to find the user within that crowd of impressions. Cellular IPs have an impression rate in the thousands (save for Verizon with hundreds), while non-cellular IPs, being more specific to individual users, were in the tens. In sum, devices on non-cellular ISPs can be targeted cheaply and effectively.

Geocoodinates. IP targeting only works well if the device connects through a consistent address, however, *geofencing* can boost recall if the device uses another IP but remains in the same general area. The downside of geographic targets are that they are less specific, thus more expensive. The reported geocoordinates are provided by the DSP, but its original source could either be from the device (GPS or network triangulation) or inferred from IP using a service like IP2Location [11] or MaxMind [12]. We were able to identify the more accurate device-sourced locations from the number of digits of precision [21]—coordinates truncated to the thousandth digit were likely inferred from IP.

Using coordinates for the training period of the first five days, we analyzed the recall over time given different search radii. Figure 10 shows these results. With a 0 km radius (i.e. exact match), recall was 42% in the first week, falling to 27%. This is not because users are returning to the exact same location many times; rather, they are connecting to the same IP, which geolocated to the same coordinates. Conversely, looking only at the 0 km recall for precise geocoordinate retargeting, we can see that devices are not often in the exact same position. Using a radius of 1 km dramatically increases recall to 67% for all GPS impressions, and 82% for precise GPS impressions. This confirms that users and their devices usually return to the same place; compared to locations reported during the first five training days, devices regularly continued to appear in the following month within 1 km.

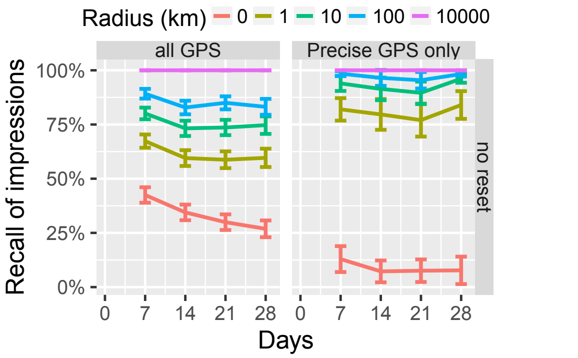

Figure 10: Recall of GPS given different resetting policies. The left fact includes geolocation and the right includes precise GPS locations only.

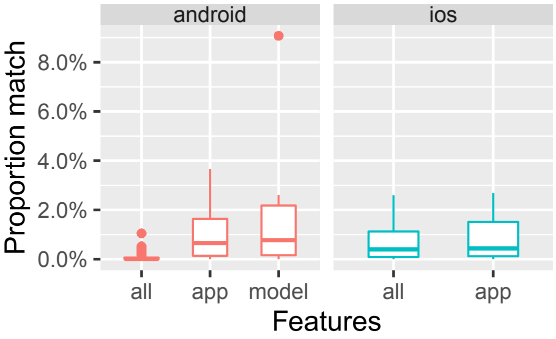

Figure 11: Average number of impressions matching each static feature. The iOS model is always reported as "iPhone", so it is not shown here.

To estimate the overhead, we used our GPS data set, which targeted areas with a 10 km radius in ten population centers in the US. We compared the resulting impression rates with measurements from the Gridded Population of the World, v4.11 data set [2], which contains population estimates within 1 × 1 km cells. Figure 12 shows these measurements; we verified that impression rate was correlated with known population (r=0.93, p<0.0001).

Additional filtering using device attributes. Our advertiser further minimizes the cost of a campaign by filtering out devices that do not match the OS, model, and app used by the target. These device attributes are unchanging, and we found that most users are found on 2 apps, with 89.2% on their favorite app more than half the time. We did not evaluate using OS version, which may

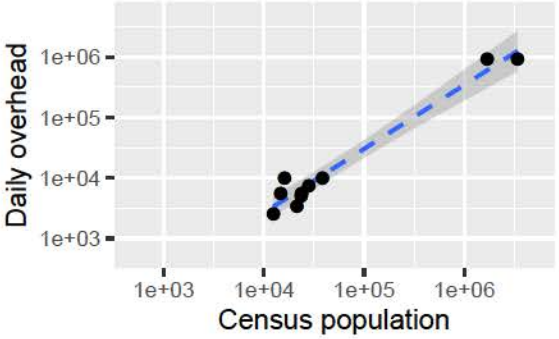

Figure 12: The impression rate of our ten target locations is correlated with the census-reported populations of the areas (r=0.93, p<0.0001) on a log-log scale. The upper bound of our sample had a rate of less than 1 impression per population per day.

be discriminating, but changes over time. We demonstrate that filtering out irrelevant devices based on these static attributes can reduce cost by several orders of magnitude.

Figure 11 shows the proportion of matching devices of each feature. Because there are a wide variety of Android devices, model is a high entropy feature for that OS. However, all iPhones self-report as "iPhone", making the iOS filter nine times less effective. Overall, the impression rates on Android devices is reduced to 0.01% using a static filter, compared to the rate without a filter; and 0.39% for iOS devices.

5.4 Cost

Cost is an important factor in RTB retargeting; campaigns can range from a few cents to millions of dollars, depending on the target filter. We evaluated the combined strategy of targeting the IP addresses and GPS coordinates associated with a device, and filtering out non-matching device models and apps. We used our `LONG` data combined with measurements from `IP` and `GPS` to perform *post hoc* analyses and quantify the effect of cost on recall. To compute recall of a device and filter, we compared the number of impressions that both matched the filter and had a retrievable mark, and compared it to the total impressions seen from that Ad ID. We developed filters for each device using impressions during the five day training period, and tested the re-identification strategy during the following month.

Impression rates during training were determined using the `IP` and `GPS` data sets, which targeted addresses and coordinates from the first five days of impressions for each device, and reduced depending on our measurements of the popularity of the device model and app. We determined costs using these rates, multiplied by the effective CPM of our impressions which was $4.02. This CPM is very high for RTB banner advertisements, but it was intentional to maximize recall—bidding optimization could lower costs in many cases.

If the Ad ID is not available, the cost to retarget a user depends on IP and geo-coordinate filters. As the advertiser's search radius expands, impressions are shown to more devices, increasing both the chances of recapturing a device and the cost of the campaign. We evaluated the cost and recall of several scenarios, which are shown in Figure 13. In each scenario, the target GPS radius increased from zero to 512 km, with the cost monotonically increasing with radius. This cost analysis is based on estimates of total impression rates from various search areas, device models, and apps in our data sets (cf. Figures 11 and 12). In general, targeting IP address alone results in at least 43% recall, and GPS alone results in at least 36% recall for an 8 km search radius. Combining IP and GPS targeting, this strategy achieved 49% recall on iOS, and 59% on Android, for less than $5/day. Without IP, performance dropped to 29% and 42% respectively. On the other hand, precise GPS devices had greater than 70% recall for $2/day, regardless of IP targeting. Far right points (which match the entire US) resulted in recalls of 68% and 76% on iOS and Android, but these tracking efforts would cost thousands of dollars per day. Android retargeting was substantially cheaper because of the higher entropy device model targeting. These results highlight that shared geolocation or IP information can significantly reduce the cost of the tracking method; while recall is still possible when both are hidden, it is more expensive.

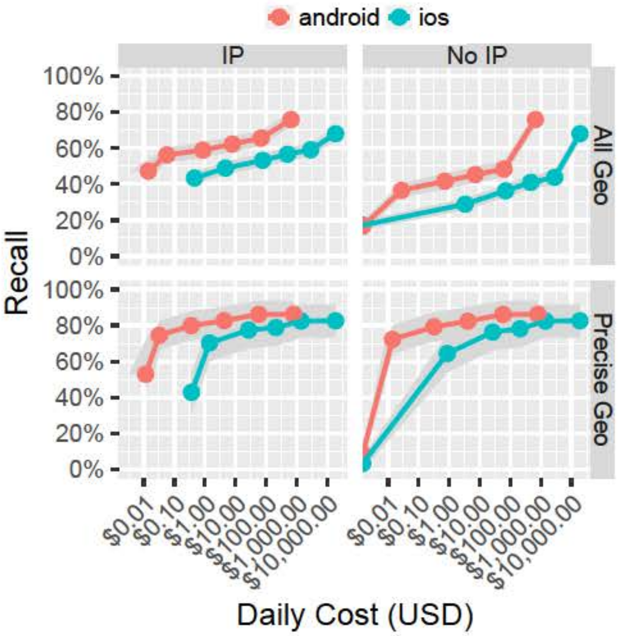

Figure 13: Recall over one month of testing, against the daily cost of targeting a device. We tested users with and without using IP address for retargeting. We separately tested users on GPS apps. We used search radii of 0 (exact GPS match), 1, 8, 64, and 512 km; the costs of these monotonically increase with search radii. We also tested without a GPS filter (far right points that are most expensive). Shaded regions represent error. Daily costs were determined by multiplying impression rate by cost per impression ($4.02 CPM, or $0.00402 per impression).

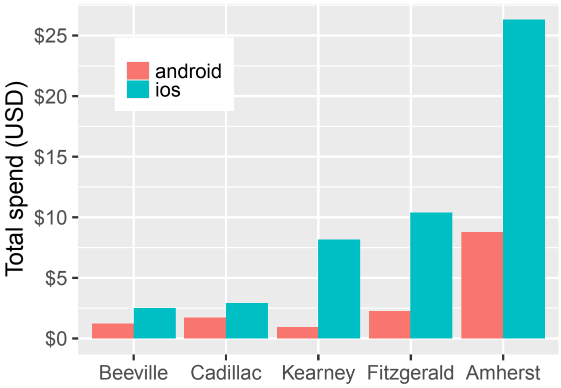

Figure 14: The total spend of our campaign in each city. Overall, we successfully rediscovered 660 devices out of 1,727 in 48 hours, including 288 that never reported an Ad ID.

6 CASE STUDIES

To provide further insight into the (1) cost of rediscovering a user without Ad ID, and (2) the effectiveness of users' privacy decisions, we conducted additional studies.

In our first case study, we deployed retargeting campaigns to rediscover devices based on single impressions, and directly measured the total cost. In our second case study, we studied a population of VPN users, and determined the ability for an advertiser to find them on a non-VPN IP address.

6.1 Case Study 1: Rediscovering Devices with One Impression

As a real-world test, we attempted to re-identify devices using a single impression and without targeting Ad ID. Our experiments above are based on a post hoc analysis of what the adversary might have achieved. Here, we carry out the tracking method rather than estimate its success and cost. This case study differs from from the LONG data set in two ways: (1) impressions were obtained using retargeting filters, instead of from Ad ID targeting with scenarios evaluated *post hoc*, and (2) we use one impression only. Since we do not target Ad ID, we simply aim to re-discover as many devices as possible while minimizing costs. While we do know that re-discovered devices are in fact the ones we targeted, we could not measure the recall as compared to targeting and Ad ID; however, this scenario allowed us to directly measure cost rather than estimate it.

We targeted five different towns and obtained a sample of 100 *device-app combinations* in each; for example, Words with Friends on Samsung Galaxy S9+. We targeted these 500 combinations for one day, within a 10 km radius from a point within each town, and used the DSP's impression capping feature to limit one impression per device. (Impression capping sometimes works for Limit At-Tracking (LAT) enabled devices; some advertising SDKs assign a proprietary temporary identifier for the purpose of per-device rate limiting.) This campaign resulted in 2,204 impressions; using our marking methods, we determined that these actually came from at most 1,727 unique devices. Of these devices, 686 (40%) hid their Ad ID. This proportion is higher than known proportions of devices with LAT enabled [7, 14], but we have found that some apps protect this information on the user's behalf, or require their own explicit agreement with the user before revealing such information to the advertiser.

We subsequently targeted the same five towns with the same combinations, except with a larger radius of 50 km to account for the possibility of users moving outside the original search radius. These were carried out in 10 campaigns (5 towns × 2 OSes), with all target apps and devices whitelisted within those campaigns. More specific targeting is possible (e.g., match app AND device rather than app OR device). In a concurrent campaign, we targeted all of the non-cellular IP addresses from the 1,727 devices. Overall, the cost of the retargeting campaigns totaled $86.73. Figure 14 shows the results on a per-city basis. The IP campaign cost $21.65, the iOS campaigns cost $50.23, and Android campaigns cost $14.85. The disparity is due in large part to the inability to target specific iPhone models.

Within 48 hours, we successfully rediscovered 660 (38%) of devices using mark and retarget, 433 of them using only geotargeting and not IP address. Among the rediscovered devices were 288 that never reported an Ad ID. This result highlights the probable ineffectiveness of iOS 14's user opt-in policy for Ad ID tracking.

6.2 Case Study 2: Re-identification of VPN Users

BitTorrent apps are well known to be a vehicle for trading copyrighted and illicit content [27, 35], and many BitTorrent users mask their activities behind VPNs. Commercial VPN services can be used to increase privacy and evade censorship [29]. They are sometimes marketed with an explicit "no logs policy" and a declared lack of cooperation with investigators.

In this section, we quantify the effectiveness of using RTB ads to unmask mobile devices behind VPNs, using the same strategies from the previous sections. Our goal was to determine the clearnet IP (and geographic location) of devices masked by a VPN. This modified goal is much closer to how investigators would apply RTB techniques; they would not focus on impression recall.

We evaluated the two simplest mechanisms in this case study. For each device we observed using a mobile BitTorrent app behind a VPN, we attempted to

1. find the device on a non-VPN/clearnet IP given only the *Ad ID*;
2. find the device on a non-VPN/clearnet IP given only a *geographic location*.

We targeted ads to μtorrent, BitTorrent, tTorrent and atorrent mobile apps[2]. The iOS appstore does not permit torrent applications (because they are so often used for illicit purposes), and so all devices in our study are Android. We used MaxMind to determine what IP addresses are owned by commercial VPN services.

To measure the fraction of devices first seen on a VPN that can be re-identified on the clearnet, we launched a campaign that targeted torrent apps on VPN IP addresses. Many of the impressions were on

[2] To be clear, we did not identify users beyond an Ad ID, nor did we identify what torrents were shared. We did not share any data with investigators.

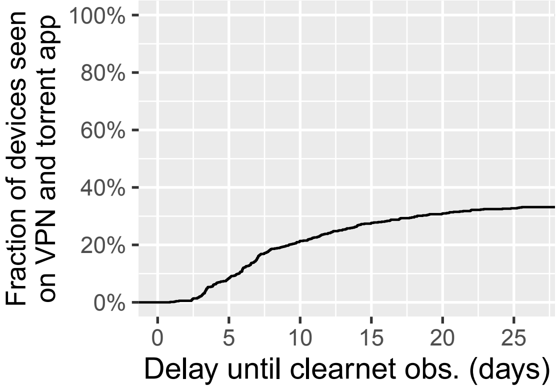

Figure 15: CDF of time required to find on clearnet the subset of 361 Ad IDs.

devices without an Ad ID because it was filtered out by the SSP or DSP due to a LAT-enabled flag. We then retargeted these Ad IDs (on any app) but restricted the campaign to clearnet IP addresses. We used a maximum CPM of $10, which is far above the level needed to be the highest bidder in the auction.

Clearnet Identification using Ad IDs. In the case where a BitTorrent user does not set the Limit Ad-Tracking flag, and the device shares its Ad ID, the adversary's strategy is simply patience: wait for each user to not use the VPN on any app. We report the results in two ways.

First, Figure 15 show the results for a set of 730 Android Ad IDs. To obtain the 730 Ad IDs, we ran a campaign for one day to get a list of Ad IDs to monitor for the following month. Then we looked for these Ad IDs in both the clearnet and VPN in a 25 day period. Within 25 days, 33% were seen on clearnet at least once, 16% were always on VPN, and 51% were not seen.

Second, we note that during the 25-day campaign, we saw a total of 13,778 distinct Android Ad IDs. We were able to re-identify 3,116 (23%) of the original Ad IDs. This percentage is slightly lower than results presented in Figure 15 because some Ad IDs may have been seen at the end of the campaign.

Clearnet Identification using Location. There is no readily apparent reason for torrent apps to require a user's geographic location. However, many torrent apps do request location access, and among the devices that shared their Ad ID, 68% reported precise, 6-digit GPS coordinate at least once in response to our impressions. These GPS values must be explicitly allowed by users before the app shares with the SSP, DSP, and advertisers.

The geographical information offered by VPN users is helpful in finding them again on the clearnet. To measure the cost of making use of this information, we performed two experiments.

First, we performed a *post hoc* analysis of geofenced retargeting of 68% of Ad IDs seen on the VPN. For each Ad ID, we determined the maximum radius of a geofence that would find and re-identify the same device on a clearnet IP address. This result is shown in Figure 16. About 30% of the devices can be located within 10 meters of the location they reported while on the VPN; and about 60% within 1 km.

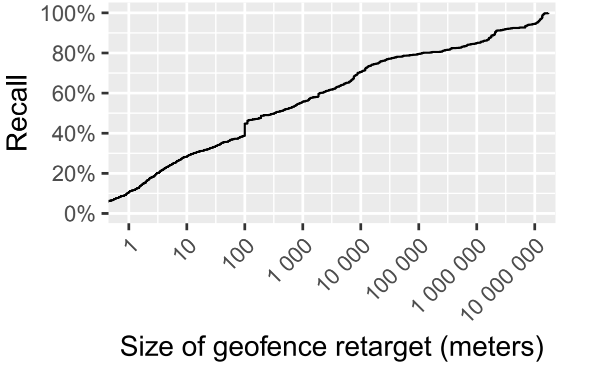

Figure 16: The maximum radius of a geofenced re-target to determine the clearnet IP address of a device seen torrenting on a VPN and sharing their GPS coordinates.

Second, we ran a live experiment to get a sense of the real costs. We operated a campaign without Ad ID restrictions for four days that targeted BitTorrent apps in New York and Los Angeles. Of the 13,778 Ad IDs, 649 shared GPS locations within 30 km of the two cities. By spending about $20 on ads over the 4 days, we were able to re-discover 64 of the 649 Ad IDs.

7 DISCUSSION

Proposed privacy-preserving solution. To defend against mark and retarget, all web storage and caches that are used for advertising should be disabled when Limit Ad-Tracking is enabled. The cost to the users would be that some ads may be downloaded more than once.

Future OS updates could drastically improve user privacy. In iOS 14, Apple devices will require users to opt in before an Ad ID is shared with anyone; this prevents advertisers from trivially tracking devices, but does not mitigate the principal tracking method studied in this paper. If all storage and caching methods are disabled, advertisers could only track users with fingerprinting techniques. These methods are stochastic, and do not guarantee that the device is the same.

Obfuscating the IP address and precise GPS location will reduce the risk of compromising an Ad ID. To prevent precise location targeting, one could use a differentially private [16] obfuscation to continue allowing advertisers to target location while making it more difficult to target single people.

Existing conditions that preserve privacy. Some advertising networks take actions that protect the privacy of users. For example, we encountered SSPs who ran JavaScript to pre-render an ad-image on a server and overlay that with a watermark; while this was likely not done specifically to protect user privacy, it did prevent users from being properly marked. We were able to detect instances of this happening by looking at the IP address, exchange, marks, and Ad ID.

Apps and SSPs also try to protect the user to some degree. Some SSPs mask the last octet of the IP address to protect the user. While this alone does not prevent user identification, it does make users more expensive to target. Most apps, even ones that request location permission, do not pass precise location on to advertisers. Some apps require users' explicit consent for personalized advertising before passing the Ad ID to advertisers.

Finally, it is more expensive for the advertiser to recall a specific iOS device, compared with Android. This disparity is due primarily to the fact that Android devices report specific model names in the user agent string, while iOS devices reveal only iPhone or iPad.

Recommendations for users to preserve privacy in the existing framework. In current systems, users can reset their identifiers manually on a regular basis, and disable it if possible. In order to minimize the efficacy of ad tracking, users should reset Ad IDs, app cache, and app data at the same time, on a regular basis. IP address was the most effective targeting feature. To prevent easy retargeting, users could browse through the cellular network, or use a VPN over WiFi, though it is not always practical.

Study limitations. Using a DSP allowed us to perform evaluations from the perspective of an advertiser. This means we are prone to variations in quality of service. JavaScript sometimes fails, or the user may exit an app while the ad is loading. Additionally, using a DSP to conduct this study allowed us a holistic view of the tracking method, but this data is collected through a third party and there are no guarantees of randomness; we balance out known biases by collecting data using a variety of campaign parameters.

For marking, we do not look at stochastic methods. Bytecode caches store the parsed JavaScript to enable faster processing. It may be possible to infer the load time of a script to see if it has been previously compiled and cached [34]. We did not pursue any user inference methods (such as IP, usage patterns, etc.) to verify user identity, since marking was highly effective, and users generally used few apps. However, these techniques could allow advertisers to track users *between* apps, without an Ad ID.

Our measurements will not necessarily produce the same results in every ad platform. Our claim is rather a demonstration that re-identification is possible to achieve relatively successfully and inexpensively. Further, in our experience, the success was repeatable and consistent.

Human Subjects. The protocols used for gathering data were reviewed and approved by our Institutional Review Board, under protocols 2016-3112 and 2016-3141. The IRB determined that the lack of PII in the study, the difficulty in obtaining informed consent, and the low risk of harm to subjects, did not necessitate obtaining informed consent.

8 RELATED WORK

Our work is related to several papers on RTB networks. None examine the accuracy of retargeting devices using marks (only Ad IDs) and none examine devices masked by VPNs.

Corner et al. [21] quantified the accessibility over time of devices to RTB advertisers, and measured features such as IP-based geolocation accuracy and bandwidth. They found that at least 14% of the population were accessible within a geo-targeted area, and found that IP-based geolocation services are generally inaccurate. (Our retargeting methods rely on geolocation consistency, rather than accuracy.) They examined device fingerprints based on features, which resulted in an accuracy of less than 50%, even if IP address is provided; our method results in zero false positives. Corner and Levine [20] explore the mobile advertising as a means of large scale scientific experimentation, quantifying the availability of sensors, and the willingness of users to click an ad to participate in an experiment.

Vines et al. [36] estimated the cost to track a single user using its Ad ID. They used a DSP to track users' specific locations, behaviours, and routines. They tracked ten devices, measuring win rates, location targeting accuracy, and found that GPS (not IP) geolocation targeting was accurate and had a low update latency. Their study also included a survey of 21 DSPs and their features; the majority of them supported HTML ads, and device and location targeting. In contrast, our study quantifies the overhead cost of tracking when an Ad ID is unavailable or reset. Also related is work by Olejnik et al. [30], who measured the cost to reveal users' browsing history through ad placements, and Bashir et al. [18] who gathered 35,000 impressions and modeled information flow between ad exchanges. Both studies quantify browsing privacy lost to advertising on websites; in contrast, our focus is on mobile apps.

In 2010, Eckersley [23] performed the first large-scale study of browser uniqueness using Java and Flash fingerprinting. However, these features are not available in mobile ads, and are largely obsolete in modern browsers. The marking aspect of this work is closely related to research in *evercookies* [26] and browser fingerprinting techniques. Gomez-Boix et al. [24] looked at many of these browser features, including font lists, canvas fingerprinting, and user-agent, and found that 33.6% of browsers were unique among 2 million users. Cao et al. [19] looked at canvas features and re-identified 1900 users with 99% accuracy. While these features are more available, they rely on hardware uniqueness and do not work well on mobile devices; mobile device hardware is much more homogeneous than computer hardware.

Phone fingerprinting has been studied to a limited extent, mostly at a smaller scale. Some studies look at fingerprinting techniques available to web and app developers that use additional device information or sensors. Das et al. [22] look at accelerometer and gyroscope and find some entropy among 63 mobile devices. These features are based on fine hardware differences between the inertial measurement units on devices; we were not able to replicate the results in preliminary studies using ads. Zimmeck et al. [37] look at cross-device tracking using 126 user-disclosed web histories and IP addresses, showing that both history and IP are effective in linking users' mobile and desktop usage. Finally, Smith et al. [34] look at bytecache timing attacks to sniff the history of a device. We could not replicate these results using ads due to the possibility of getting banned by the DSP.

Work by Ikram et al. [25] and Perta et al. [32] on security for VPN users is complementary to ours. Both focused on vulnerabilities and exposures present in the client software itself, such as the presence of malware and leakage of IPv6 traffic. Our results would hold even if the client software had no issues.

9 CONCLUSION

We demonstrated that advertisers can re-identify devices on RTB networks even when users have taken steps to prevent it, such as disabling the device Ad ID or using a VPN. We have shown that advertisers can take advantage of the permissive configuration of web views to place long-lasting marks on the device, retarget groups of devices using device profiles, and then re-identify the original device within that group.

We do not believe these results are well-known in the industry, though many tracking technologies are closely held in the competitive ad industry. We find it surprising that marking features are allowed and how difficult it is for users to take precautions to prevent reidentification in ad networks.

We find that at least 68% of the impressions from these devices can be re-identified without an Ad ID if the adversary had a large advertising budget. For less than $5 per day, we could rediscover a device with 49% success, compared with using an Ad ID, over one month. User privacy can be increased if OS developers disable web view storage by default, or give API access to disallow storage and caching within advertising web views.

This project was supported in part by Grant# 2018-MC-FX-K059 awarded by the Office of Juvenile Justice and Delinquency Prevention, Office of Justice Programs, U.S. Department of Justice. The opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect those of the Department of Justice.

APPENDIX

A AD TAG HTML

Below is a simplified excerpt of the *dynamic* ad tag we used in experiments. It contains ad *macros* which are replaced by the DSP. It also contains a call to a JavaScript file on our server, which performs the marking functions. A static ad would only include the URL of the image *pixel.gif* file, conventionally an empty 1 × 1 pixel image.

```
<!-- This ad is part of a research project at
    Location A. ... //-->
<a href="https://www.example.tld">
<img id="splash_pixel" style="display:none"
src="https://api.example.tld/pixel.gif?impression=
    ${impression_id}&device_identifier=${
    device_identifier}&${imp_id}&gps=${gps}&
    campaign=${campaign_id}..." /></a>
<script src="https://api.example.tld/mark.js">
</script>
```

B DISABLING MARKING WITHIN ANDROID WEB VIEWS

We tested the effect of different Java methods in disabling various marking strategies. We found that Android developers who want to disable all marking capabilities must periodically call WebStorage's `deleteAllData` to delete HTML5 storage data, and CookieManager's `removeAllCookies` to delete cookies [3]. They may also completely disable some, but not all marking features: WebStorage's `setDomStorageEnabled` and `setDatabaseEnabled`, and CookieManager's `setAcceptCookie` methods affect LocalStorage, Web SQL, and HTTP Cookies respectively. However, these latter methods do not affect IndexedDB, which must be cleared using `deleteAllData`. Developers on iOS may follow a similar procedure discussed in [6].

REFERENCES

[1] Advertising id. https://support.google.com/googleplay/android-developer/answer/6048248.

[2] Gridded population of the world, version 4 (gpwv4): Population count grid. http://dx.doi.org/10.7927/H4X63JVC. Accessed: 2019-09-19.

[3] https://developer.android.com/reference/android/webkit/.

[4] https://developer.apple.com/app-store/user-privacy-and-data-use/.

[5] https://developer.chrome.com/apps/offline_storage. Accessed: 2020-03-24.

[6] https://stackoverflow.com/questions/8500334/how-to-remove-html5-local-storage-of-an-ios-app-using-uiwebview.

[7] https://support.appsflyer.com/hc/en-us/articles/115003734626-FAQ-Impact-of-Apple-Limit-Ad-Tracking-on-attribution.

[8] https://support.appsflyer.com/hc/en-us/articles/115003734626-FAQ-Impact-of-Apple-Limit-Ad-Tracking-on-attribution.

[9] https://www.adpushup.com/blog/explainer-the-four-types-of-programmatic-deals/.

[10] https://www.census.gov/data/tables/time-series/demo/popest/2010s-total-metro-and-micro-statistical-areas.html#par_textimage_1139876276.

[11] https://www.ip2location.com/. Accessed: 2020-03-24.

[12] https://www.maxmind.com/. Accessed: 2020-03-24.

[13] https://www.mediapost.com/publications/article/341573/83-days-until-christmas-when-will-marketers-spe.html.

[14] https://www.verve.com/limit-ad-tracking/.

[15] https://www.visualcapitalist.com/the-covid-19-impact-on-advertising-spend/.

[16] Andrés, M. E., Bordenabe, N. E., Chatzikokolakis, K., and Palamidessi, C. Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security* (2013), ACM, pp. 901–914.

[17] Balakrishnan, M., Mohomed, I., and Ramasubramanian, V. Where's that phone? Geolocating IP addresses on 3G networks. In *ACM IMC* (2009), pp. 294–300.

[18] Bashir, M. A., Arshad, S., Robertson, W., and Wilson, C. Tracing information flows between ad exchanges using retargeted ads. In *USENIX Security Symposium* (2016), pp. 481–496.

[19] Cao, Y., Li, S., Wijmans, E., et al. (cross-) browser fingerprinting via os and hardware level features. In *NDSS* (2017).

[20] Corner, M. D., and Levine, B. N. Micromobile: Leveraging mobile advertising for large-scale experimentation. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services* (2018), ACM, pp. 310–322.

[21] Corner, M. D., Levine, B. N., Ismail, O., and Upreti, A. Advertising-based measurement: A platform of 7 billion mobile devices. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking* (2017), ACM, pp. 435–447.

[22] Das, A., Borisov, N., and Caesar, M. Tracking mobile web users through motion sensors: Attacks and defenses. In *NDSS* (2016).

[23] Eckersley, P. How unique is your web browser? In *International Symposium on Privacy Enhancing Technologies Symposium* (2010), Springer, pp. 1–18.

[24] Gómez-Boix, A., Laperdrix, P., and Baudry, B. Hiding in the crowd: an analysis of the effectiveness of browser fingerprinting at large scale. In *Proceedings of the 2018 World Wide Web Conference* (2018), International World Wide Web Conferences Steering Committee, pp. 309–318.

[25] Ikram, M., Vallina-Rodriguez, N., Seneviratne, S., Kaafar, M. A., and Paxson, V. An analysis of the privacy and security risks of android vpn permission-enabled apps. In *Proceedings of the 2016 Internet Measurement Conference* (New York, NY, USA, 2016), IMC '16, Association for Computing Machinery, pp. 349–364.

[26] Kamkar, S. evercookie.

[27] Kigerl, A. C. Infringing nations: Predicting software piracy rates, bittorrent tracker hosting, and p2p file sharing client downloads between countries. *International Journal of Cyber Criminology 7*, 1 (2013), 62.

[28] Mozilla. Apple: Rotate tracking ids on iphone each month. https://foundation.mozilla.org/en/campaigns/privacy-thats-iphone-but-is-it/.

[29] Ndhlovu, L. Facing internet restrictions, journalists turn to vpns. https://ijnet.org/en/story/facing-internet-restrictions-journalists-turn-vpns Accessed: 2020-03-24.

[30] Olejnik, L., Minh-Dung, T., and Castelluccia, C. Selling off privacy at auction.

[31] Open Standards for Real-Time Bidding (RTB). OpenRTB Mobile RTB API v1.0, Feb 2011.

[32] Perta, V. C., Barbera, M. V., Tyson, G., Haddadi, H., and Mei., A. A glance through the vpn looking glass: Ipv6 leakage and dns hijacking in commercial vpn clients. In *Proc. PETS* (2015).

[33] Raghavan, B., Kohno, T., Snoeren, A. C., and Wetherall, D. Enlisting isps to improve online privacy: Ip address mixing by default. In *International Symposium on Privacy Enhancing Technologies Symposium* (2009), Springer, pp. 143–163.

[34] Smith, M., Disselkoen, C., Narayan, S., Brown, F., and Stefan, D. Browser history re: visited. In *12th {USENIX} Workshop on Offensive Technologies ({WOOT} 18)* (2018).

[35] U.S. Dept. of Justice. The National Strategy for Child Exploitation Prevention and Interdiction: A Report to Congress. https://www.justice.gov/psc/file/842411/download, April 2016.

[36] Vines, P., Roesner, F., and Kohno, T. Exploring adint: Using ad targeting for surveillance on a budget-or-how alice can buy ads to track bob. In *Proceedings of the 2017 on Workshop on Privacy in the Electronic Society* (2017), ACM, pp. 153–164.

[37] Zimmeck, S., Li, J. S., Kim, H., Bellovin, S. M., and Jebara, T. A privacy analysis of cross-device tracking. In *26th {USENIX} Security Symposium ({USENIX} Security 17)* (2017), pp. 1391–1408.