Measuring and Fingerprinting Click-Spam in Ad Networks

Vacha Dave [*]
MSR India and UT Austin
vacha@cs.utexas.edu

Saikat Guha
Microsoft Research India
saikat@microsoft.com

Yin Zhang
Univ. of Texas at Austin
yzhang@cs.utexas.edu

ABSTRACT

Advertising plays a vital role in supporting free websites and smartphone apps. Click-spam, i.e., fraudulent or invalid clicks on online ads where the user has no actual interest in the advertiser's site, results in advertising revenue being misappropriated by click-spammers. While ad networks take active measures to block click-spam today, the effectiveness of these measures is largely unknown. Moreover, advertisers and third parties have no way of independently estimating or defending against click-spam.

In this paper, we take the first systematic look at click-spam. We propose the first methodology for advertisers to independently measure click-spam rates on their ads. We also develop an automated methodology for ad networks to proactively detect different simultaneous click-spam attacks. We validate both methodologies using data from major ad networks. We then conduct a large-scale measurement study of click-spam across ten major ad networks and four types of ads. In the process, we identify and perform in-depth analysis on seven ongoing click-spam attacks not blocked by major ad networks at the time of this writing. Our findings highlight the severity of the click-spam problem, especially for mobile ads.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Online Advertising Fraud*

Keywords

Click-Spam, Click-Fraud, Invalid Clicks, Traffic Quality

1. INTRODUCTION

Background and motivation: Click-spam costs online advertisers on the order of hundreds of millions of dollars each year [4]. Instead of supporting free smartphone apps and websites, this money ends up in the pocket of click-spammers. Click-spam[1] subsumes a number of scenarios that all have two things in common: (1) the advertiser is charged for a click, and (2) the user delivered to the ad's target URL has no actual interest in being there. Click-spam can be generated using a variety of approaches, such as (i) botnets (where malware on the user's computer clicks on ads in the background), (ii) tricking or confusing users into clicking ads (e.g., on parked domains), and (iii) directly paying users to click on ads.

[*]Work done while at Microsoft Research India

[1]or equivalently, click-fraud. Since a fraudulent motive is often difficult to conclusively prove, we use the term click-spam instead.

Incentives for click-spam are linked directly to the flow of money in online advertising — advertisers pay ad networks for each click on their ad, and ad networks pay publishers (websites or phone apps that show ads) a fraction (typically around 70% [2]) of the revenue for each ad clicked on their website or app. A publisher stands to profit by attracting click-spam to his site/app. An advertiser stands to inflict losses on his competitor(s) by attracting click-spam to his competitors' ads. An advertising network stands to increase revenues (but lose reputation) by not blocking click-spam.

Reputed online ad networks have in-house heuristics to detect click-spam and discount these clicks [39]. No heuristic is perfect. Advertisers pay for false negatives (click-spam missed by the heuristic). None of the ad networks we checked release any specifics about click-spam (e.g., which keywords attract click-spam, which clicks are click-spam, etc.) that would otherwise allow advertisers to optimize their campaigns, or compare ad networks.

Research goals and approach: We have two main research goals in this paper. Our primary goal is to design a methodology that enables advertisers to independently measure and compare click-spam across ad networks. The basic idea behind our approach is simple: since the user associated with click-spam is, by definition, not interested in the ad, he would be less likely to make any extra effort to reach the target website than a user legitimately interested in the ad. The advertiser can measure this difference and use it to estimate the click-spam fraction. Of course some legitimately interested users may not make the extra effort (false positives), or some uninterested users may still make the extra effort (false negatives). We correct for both types of errors by using a Bayesian framework, and by performing experiments relative to control experiments. Section 3 details our methodology.

Validating the correctness of our methodology is challenging because there is no ground truth to compare against. Ad networks do not know the false negative rate of their heuristics, and thus tend to underestimate click-spam on their network [12]. The accuracy of heuristics used by third-party analytics companies (e.g., Adometry) is unknown since their methodology and models are not open to public scrutiny; indeed, ad networks contend they overestimate click-spam [16]. We manually investigate tens of thousands of clicks we received (in Section 5). We present incontrovertible evidence of dubious behavior for around half of the search ad clicks and a third of the mobile ad clicks we suspect to be click-spam, and circumstantial evidence for the rest, thus establishing a tight margin-of-error in our methodology. In the process, we discover seven ongoing click-spam attacks not currently caught by major ad networks, which we reported to the parties concerned.

Our secondary goal is to measure the magnitude of the click-spam problem today. To this end, in Section 4, we apply our methodology to measure click-spam rates across ten major ad networks (including Google, Bing, AdMob, and Facebook) and four ad types. Our work represents the first measurement study of click-spam.

We also identify key research problems that can have a measurable impact in tackling click-spam.

Contributions: We make three main contributions in this paper. (i) We devise the first methodology that allows advertisers

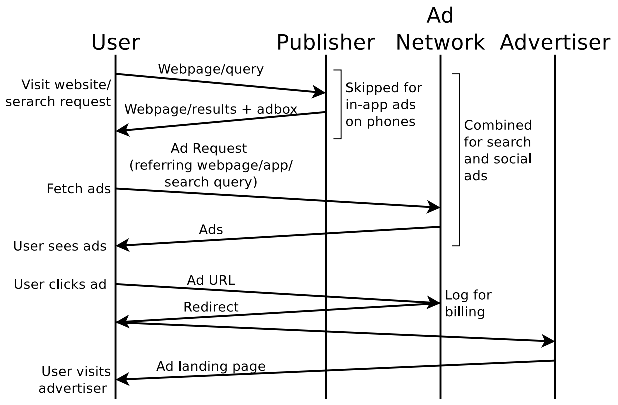


Figure 1: Time-line for serving ads.

to independently measure and compare click-spam rates across ad networks. We validate the correctness of the methodology using real-world data. (ii) We report on the sophistication of ongoing click-spam attacks and present strategies for ad networks to mitigate them. (iii) We conduct a large-scale in-depth measurement study of click-spam today.

2. ONLINE ADVERTISING PRIMER

Search vs. contextual: Keyword-based advertising is broadly classified into two categories: (i) *search advertising*, which are ads based on search keywords that show up on the side of search results, and (ii) *contextual advertising*, which are ads that show up on Web pages or in applications based on keywords extracted from the context. Search ads may be *syndicated* — *i.e.*, they are shown not only on the search engine operated by the ad network (*e.g.*, on www.google.com), but also on affiliate websites that offer customized search engines (*e.g.*, www.ask.com). The term *publisher* refers to the party that showed the ad (*e.g.*, website for contextual ads, smartphone application for mobile ads, affiliate for syndicated search ads). We do not consider other kinds of ads (*e.g.*, ads in videos, banner ads, etc.) in this paper.

Mobile vs. non-mobile: Both search and contextual advertising can be further classified as mobile or non-mobile based on what device the search or webpage request originated from. Mobile includes smart-phones and other mobile devices that have "full browser capabilities", as well as feature-phones with limited WAP browsers. The reason we draw a distinction between mobile and non-mobile is because we found ad networks internally seem to have very different systems for serving the same ads to mobile vs. to non-mobile users. We do not know the reason for this difference, but speculate it is because ad networks tend to expand through mergers and acquisitions, resulting in multiple technology stacks operating concurrently.

Ad delivery: Figure 1 illustrates the time-line for serving online ads. When the user visits a publisher website, the website returns an adbox (*e.g.*, embedded iframe), which causes the user's browser to contact the ad network. The request to the ad network identifies the referring website through the HTTP Referer (sic) header. The ad network then populates the adbox with contextual ads. In an alternate mechanism (not shown), premium publishers may directly query the ad network for relevant ads and seamlessly integrate them into the website content.

Charging model: While there are multiple charging models for online advertising (*e.g.*, impression-based, action-based), by far the most common is pay-per-click (PPC or CPC), where the advertiser is charged (and publisher paid) only if the user clicks on the ad. The publisher gets some fraction (typically 70%) of the revenue that the ad network collects from the advertiser. The accounting is performed as follows. The ad URL points to the ad network's domain with information about which ad was clicked (encoded in the GET parameters). When the user clicks an ad, the browser contacts the ad network, which logs the encoded information for billing purposes and redirects the user to the advertiser's site. This redirection is typically performed through an HTTP 302 response, which preserves the publisher's URL in the HTTP Referer seen by the advertiser's Web server. Black-hat techniques such as Referer-Cloaking [5] by the publisher, ad network policies [14], or bugs in browsers and proxies may result in empty or bogus Referer values being sent to the advertiser.

User engagement: The ad network can track limited user engagement (*i.e.*, ads viewed or clicked) for multiple ads shown across multiple publishers (*e.g.*, using cookies), but cannot, in general, track user engagement after the click. The advertiser, on the other hand, can track detailed user actions (only) on the advertiser's own website, but cannot track user engagement with other ads. Thus while the ad network has a broad-but-shallow view, the advertiser has a narrow-but-deep view into user engagement.

Click-spam discounts: Ad networks internally discount clicks based on in-house heuristics. The user is still redirected to the advertiser, but the advertiser is not charged for the click. Ad networks *do not* indicate which clicks were charged and which not in the advertiser's billing report.

3. ESTIMATING CLICK-SPAM

In this section we design a method that any party (*e.g.*, advertisers, ad agency, or researchers) can use to estimate click-spam rates for a given ad without explicit cooperation from the ad network.

3.1 Challenges

No ground truth: It is not possible to first identify (definitively) which clicks are click-spam, and then compute what fraction of the total traffic click-spam accounts for. A click is click-spam if the user did not *intend* to click the ad. There is no way to conclusively determine user intent without explicitly asking the user.

No global view: As mentioned, the ad network cannot track user engagement on the advertiser site, and the advertiser has no knowledge of the user's engagement with other advertisers. For example, the ad network does not know if the user never loads the advertiser's site after the click (we saw botnets exhibiting this behavior), and the advertiser does not know if the same user is implicated in click-spam attacks on another advertiser. The obvious solution is for the ad network and advertiser to cooperate. But financial disincentives and legal concerns prevent them from doing so: the advertiser could lie in his favor (claiming fraud where there is none) in order to gain deeper discounts from the ad network; the ad network could be held liable if sharing user history with advertisers has unforeseen privacy consequences.

Granularity: The granularity over which click-spam is estimated is important. An ad network may have low click-spam overall, say, but certain lucrative segments (*e.g.*, mortgage) may be experiencing orders of magnitude more click-spam. Advertisers require fine-grained measurements for their selected set of keywords.

Noise: As in any Internet-scale system, data is extremely noisy. We encountered users where Referer headers are inexplicably omitted, or browser User-Agents, IP addresses, cookies etc. change inexplicably within the same session (perhaps a buggy browser or proxy), and bad publishers that behave non-deterministically (perhaps to avoid detection). At the same time, because clicking an ad is a rare event, gathering good data is time-consuming. The combination results in very low signal-to-noise ratios.

3.2 Our Approach

We apply a Bayesian approach to work around the lack of ground truth. Instead of attempting to conclusively identify which clicks are click-spam, for a given ad, we create two scenarios detailed below where the (unknown) fractions of click-spam traffic is *different*. We link the two using a Bayesian formula to effectively cancel out quantities we cannot measure. The remaining quantities are those an advertiser can measure locally without requiring a global view. We control for noisy data (*i.e.*, adjust for false-positives and false-negatives) using a control experiment. Our approach *does not* (and indeed cannot) report whether a specific click is click-spam or not. The output is a single number representing our estimate of the fraction of clicks *for the given ad* that click-spam accounts for.

3.2.1 Data Collection

The detailed data collection procedure is as follows.

1. The advertiser (or a researcher signed-up as an advertiser with an ad network) creates his ads of interest in the usual way. That is, enters the text of the ad, selects ad targeting criteria (*i.e.*, search keywords, user demographics, mobile vs. non-mobile, etc.), sets his advertising budget (*i.e.*, auction bid amount, daily-budget), and sets the destination URL of the ad to his *landing-page* — a page on the advertiser's website the user should be sent to.

2. When a user clicks the ad, the advertiser website either loads the intended landing-page for the ad, or with some (small) probability first loads an interstitial page before eventually loading the landing-page. The point of the interstitial page is to *turn away* some fraction of users (legit clicks or not). To this end, the interstitial page may require some passive or active user engagement before continuing on to the landing-page. An example of passive engagement is to require the user to stay on the page for a few seconds before continuing on to the landing page (the interstitial page in this case may simply show a "loading..." message; it is known that a few seconds increase in page loading time can result in a measurable drop in traffic [6]). Some examples of active engagement is to require the user to click a link (*e.g.*, "This page has moved. Click here to continue.") or, in the extreme case, solve a CAPTCHA [15], both of which also result in significant drops in traffic.

Assumption 1. An implicit assumption is that the interstitial page will turn away a (unknown) larger portion of the click-spam traffic than the (unknown) portion of legitimate traffic it turns away. Specifically, we *do not assume* for example that all good users will patiently wait a few seconds; rather we assume that a smaller portion of click-spam traffic will wait the duration than the portion of good users doing so. Indeed user dwell time and willingness to click have long been considered an implicit measure of user interest [29, 35]. Interstitial pages leverage this knowledge to enhance the quality of the traffic reaching the landing-page by some (unknown) amount as compared to without the interstitial page.

3. Next the advertiser defines some advertiser-specific user engagement that he considers ultimate proof that the user intended to click the ad. For example, if the user makes a financial transaction on the advertiser's site, or signs up for some mailing list, etc. We call such users *gold-standard* users. Certainly not all users are gold-standard. But as long as a handful of users coming directly to the landing-page, and a handful coming to the landing-page via the interstitial page are gold-standard, we can apply our Bayesian approach below.

Assumption 2. We make an implicit assumption that for users that are *not turned away* by the interstitial page, their likelihood of becoming gold-standard users is not significantly changed. We *do not assume* that the interstitial page will not turn away users who may have become gold-standard users. Rather we assume for example that some users will not wait on the interstitial page, but if they do, they will not hold a grudge. We empirically find this assumption to hold in practice as we report later.

4. Lastly, extending the technique in [25], the advertiser creates a second ad that is identical to the ad created in step 1 above with the exception of the text of the ad. That is, the second ad has the same targeting criteria, advertising budget, and destination URL, but the text of the ad is junk (*e.g.*, a random nonsensical combination of words). We use this ad to adjust for false positives.

Assumption 3. We assume that very few users will intentionally click the second ad. While some users may be curious about the random set of words, we find this assumption is borne out in practice. We also assume for now that the click-spam *click-through-ratio i.e.*, the ratio of click-spam clicks to impressions of the ad) is independent of the text of the ad (we relax this assumption later).

3.2.2 Bayesian Estimation

Let G_d and G_i be the event that the user is a gold-standard user that arrived either directly, or via the interstitial page respectively. Let I_d and I_i be the event that the user intended to click the ad (*i.e.*, not click-spam) out of all users directly reaching the landing-page, or all users reaching via the interstitial page respectively.

Bayesian equation for $P(I_d)$: The advertiser is interested in learning $P(I_d)$. G and I are linked using Bayes theorem as follows: $P(G|I) = P(I|G) \times P(G)/P(I)$. Note that a gold-standard user implies that the click is not click-spam, *i.e.*, $P(I|G) = 1$. As discussed above in Assumption 2, $P(G_d|I_d) \simeq P(G_i|I_i)$; substituting and simplifying yields: $P(I_d) = \frac{P(G_d) \times P(I_i)}{P(G_i)}$.

$P(G)$ is computed as the ratio of the number of gold-standard clicks (g; known) to the number of clicks (n; known). $P(I_i)$ is the ratio of the number of non-click-spam clicks (i_i; unknown) to the number of clicks (n_i; known) arriving via the interstitial page. The above equation reduces to:

$$P(I_d) = \frac{\frac{g_d}{n_d} \times \frac{i_i}{\cancel{n_i}}}{\frac{g_i}{\cancel{n_i}}} = \frac{g_d \times i_i}{n_d \times g_i} \qquad (1)$$

Only i_i on the right-hand-side is unknown.

Estimating i_i: As mentioned, the interstitial page enhances the traffic quality. If it did a perfect job — *i.e.*, all unintentional clicks would be turned away, and all intentional clicks would pass through — computing i_i would be trivial. In practice, the interstitial page has false-negatives (turns away users intentionally clicking the ad) and false-positives (not turning away click-spam). False-negatives do not affect i_i since it is the number of non-click-spam clicks *actually reaching* the landing-page, but false-positives result in an inflated value of i_i. We use a *control ad* to estimate the number of false-positives, and adjust for it.

Let F_i and T_i be the false-positive and true-positive click-through-ratios for the original ad and the interstitial page i. Similarly, let F_i' and T_i' be the false- and true-positive click-through-ratios for the control ad (identical ad except with junk ad text as mentioned above). We have four unknowns, and need four equations to solve. As discussed above in Assumption 3, we assume $T_i' \simeq 0$ and $F_i = F_i'$. The advertiser can measure $F_i + T_i = l_i/d$ where l_i is the number of clicks for original ad reaching the landing page through the interstitial page, and d is the number of impressions of the original ad as reported by the ad network, and the corresponding equation $F'_i + T'_i = l'_i/d'$ for the control ad.

The estimate for i_i is simply $T_i \times d$. Solving the four equations above for T_i we get the value of i_i adjusted downwards to account for false-positives as: $i_i = l_i - l'_i \times \frac{d}{d'}$

Final estimation formula: Combining the above with Eq. (1) we can estimate the click-spam rate for the original ad as:

$$P(I_d) = \frac{g_d \times \left(l_i - l'_i \times \frac{d}{d'}\right)}{n_d \times g_i} \tag{2}$$

where:

g_d, g_i	: numbers of gold-standard users arriving directly and through the interstitial page respectively
d, d'	: number of impressions of the original ad and control ad respectively
l_i, l'_i	: number of clicks reaching (via the interstitial page) the landing-page for the original ad and control ad respectively
n_d	: number of clicks on the original ad directly reaching the landing-page

All these quantities can either be measured directly by the advertiser, or are present in billing reports ad networks generate today.

3.3 Limitations

A key limitation of our approach is that the advertiser must *actively* measure click-spam. The advertiser must interpose the interstitial page on live traffic (that he has paid-for), create a control ad (that he needs to pay for) to correct for false-positives, etc. Both the interstitial ad and control ad harm the user experience. It would be far more desirable to be able to passively look at logs and be able to estimate the click-spam rate from them.

One way to minimize the user experience impact is to apply our approach reactively when click-spam is suspected, but that runs into a second limitation — the rarity of data. Any estimation technique requires statistically significant data. The crucial factor in Eq. (2) is g_i and g_d — the number of gold-standard users. If these are small, the click-spam estimate can swing wildly. Suppose the advertiser manages to identify two gold-standard users, one arriving through the interstitial page and one directly, and computes a click-spam estimate based on it. If one new gold-standard user arrives through the interstitial site (or directly), the new click-spam estimate is half the previous (or will double). For a statistically significant estimate, as we report later, the advertiser must wait for roughly 25 gold-standard users via the two paths. This is especially an issue for small advertisers. Small advertisers may have to wait a long time to get gold-standard users — low advertising budgets means their ads don't get shown as much, even if they get shown users may not click on poorly ranked ads, even if they click they may not engage in a financial transaction with the advertiser, etc. The need to gather data over such an extended period is clearly at odds with minimizing the impact on user experience.

A group of small advertisers targeting similar keywords/users (or an ad-agency representing them) can apply our approach in the aggregate. Doing so has two benefits. First, due to the aggregation effect the group accretes statistically significant data more quickly. And second, the user experience impact is amortized across many advertisers. The downside, however, is that advertisers lose the ability to individually define what a gold-standard user means (which our approach otherwise allows) and have to depend on someone other than themselves to estimate click-spam rates.

Finally, our approach is naturally sensitive to the three choices the advertiser needs to make: 1) what his definition of a gold-standard user is, 2) what interstitial page approach he wishes to use, and 3) what the text of the control ad is. We discuss the implication of each design decision in turn. First, if the advertiser sets too high a bar for the gold-standard user he may not get statistically significant data; if he sets too low a bar that even click-spam users get classified as gold-standard he will underestimate click-spam rates. Second, if the advertiser picks too easy an interstitial page (everyone gets through), in Eq. (2) g_d/g_i will approach n_d/l_i and the estimate will approach 1 (*i.e.*, all clicks are legitimate) if the advertiser doesn't use a control ad; or 0 if he uses a control ad (*i.e.*, no clicks are valid). If the advertiser picks too hard an interstitial page (no one gets through), g_i and l_i will both approach 0, and the click-spam estimate will become undefined. Thus there is clearly some sweet-spot in designing the interstitial page, which we do not discuss. Third, if the control ad is not independent of the original ad (*e.g.*, the random choice of words happens to be related to the original ad), false-positives may be over- or under- corrected for. Making the right design choices is advertiser-specific.

To address the above issues to some degree, we report in the next section our experience with multiple types of interstitial pages, different definitions and numbers of gold-standard users. While our data shows much promise in our approach, we stress that a more thorough evaluation is needed.

4. MEASURING CLICK-SPAM TODAY

In this section we first validate the correctness of our approach from the previous section. We then conduct a large-scale measurement study of ten major ad networks and four types of ads.

Validation strategy: We assume that reputed search ad networks (specifically Google and Bing) are mature enough that their in-house algorithms are able to detect and discount for most of the click-spam on their search affiliate network. Validating our measurement approach then involves computing our click-spam estimate and comparing it to the *charged clicks* for Google and Bing search ads. Note that our algorithm does not have access to any data (including historical and aggregate data) that in-house algorithms at Google and Bing have access to, and Google and Bing do not have access to the detailed user-engagement data we collect as advertisers for user clicking our ads (specifically, we do not use any of the analytics products offered by Google or Bing). Given the datasets are completely different, if the click-spam rates we compute match that computed by leading ad networks (which they do as we report below), we have a strong reason to believe that our measurement approach is sound.

4.1 Methodology

We sign-up with ad networks as three different advertisers (each targeting different keywords) and follow the methodology from the previous section. The first advertiser targets a highly popular keyword (celebrity). The second, a medium-popularity keyword (yoga). And the third, a low-popularity keyword (lawnmower). We pick the keywords from a ranked list of popular keywords that the advertising tools of these ad networks provide.

For each keyword we create a realistic looking landing page, since the policies of the ad networks require us to use keywords that are relevant to the landing page. We instrument the landing-page to track mouse-movement, time spent on the page, switching browser tabs into or away from our page, whether any link on the page was clicked or not, page scroll, etc. Our instrumentation is through Javascript on the page; we detect browsers that don't support Javascript or have it disabled and exclude those data points. We direct on-path proxies (if any) to not cache any response so our server logs accurately reflect accesses. Unless otherwise mentioned, we pick a lax definition of gold-standard users based on

Figure 2: A control ad; content is a random set of English words

this telemetry, which is, that the user stays on the page for at least five seconds, and (for non-mobile browsers) produces at least one mouse/cursor move event. We are unable to define gold-standard users based on financial transactions, even though we expect that to be very strong signal of intent for real advertisers.

Next we create three interstitial pages: the first shows a loading message for five seconds before automatically redirecting to the landing-page. The second asks the user to click a link to continue to the landing-page. And the third asks the user to solve a CAPTCHA. We do not test the CAPTCHA interstitial for Google traffic since their advertiser policies restrict us from doing so.

We then create four ads for each target landing-page. The first ad directly takes the user to the landing page. The second, third and fourth ads first take the user to the three interstitial pages respectively, before continuing on to the landing page. All ads target the same keyword(s), user demographics, device and platform types, etc. The reason we create four separate ads (instead of a single one and interposing the interstitial page after the click) is so that Google/Bing produce fine-grained billing reports and statistics for each ad, which we can then validate our design choices against.

We create four additional (control) ads for each landing-page that correspond to the four original ads, but with junk ad text. The ad text was generated by picking five random words from an English dictionary (*e.g.*, Figure 2).

4.2 Scale

We repeat the above for 10 ad networks. For search ads we measure Google Search, Bing Search, and 7Search. For contextual ads we measure Google AdSense and Bing Contextual. For mobile ads we measure Google Mobile, Bing Mobile, AdMob (now owned by Google), and InMobi. And lastly for social ads, we measure Facebook. Altogether this adds up to 216 ads across all the networks.

We run the ads for a period of 50 days as needed to gather enough data. The majority of the ads were flighted in early January 2012. We continually adjust bids (mostly revising them higher) to help the lower popularity ads quickly attract enough data.

In all our ads were shown 26M times across all ad networks. They resulted in a total of 85K clicks (17K charged). Our ads were shown at at-least 1811 publisher websites and mobile apps (but the true number is likely much higher since we cannot determine the publisher for over 65% of our traffic). The landing pages were fetched by a total of 33K unique IP addresses located in 190 countries. We encountered over 7200 browser User-Agent strings (after sanitizing them to remove browser plugin version numbers).

4.3 Data

We log all web requests made to our server. The logs used in this study are standard Apache webserver logs that include the user's IP address, date and time of access, URL accessed (of a page on our webserver) along with any GET parameters, the HTTP Referer value and User-Agent value sent for that request, and a cookie value we set the first time we see a user to identify repeat visits from the same user. The raw logs including user engagement telemetry weighs in at over 3 GB.

A sanitized version of our raw logs is available online[2].

[2]http://www.cs.utexas.edu/~vacha/sigcomm12-clickspam.tar.gz

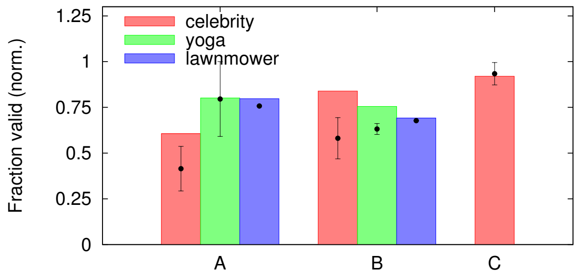


Figure 3: Normalized estimates for search ads

4.4 Ethics

Throughout our study we followed the advertiser terms-of-service (current as of when we did the measurement) for each of the ad networks we measured. Whenever our ads were rejected by the ad network (due to policy reasons) we fixed the issue so as to be compliant; if we couldn't fix it, we simply dropped that data-point.

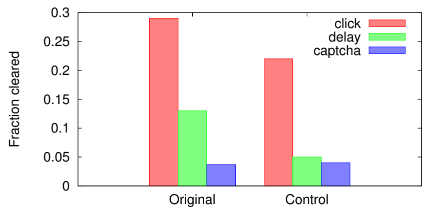
High click-spam is an embarrassment for ad networks. Our goal in this paper is to systematically design a methodology, highlight the severity of the click-spam problem, and give researchers the tools and knowledge to further the state of the art. Our goal is not to embarrass ad networks. As a result, we prefer to report normalized or relative numbers whenever possible, and anonymize ad network names whenever it does not affect the core message of this paper.

Lastly, we expressly try to minimize adversely impacting user experience on these ad networks. For example, in order to get enough clicks on an ad, we have two options: run the campaign for longer, or increase the bid amount. We always choose the latter to minimize the time our ad is active on the network. For ads where despite increasing the bid we cannot gather traffic fast enough, we prefer to give up on that data-point and stop running that ad. Minimizing the time our ads are active also minimizes our contribution to the existing auction volatility for the keywords we bid on. As of this writing we have not received any complaints from ad networks, users, or advertisers regarding our study.

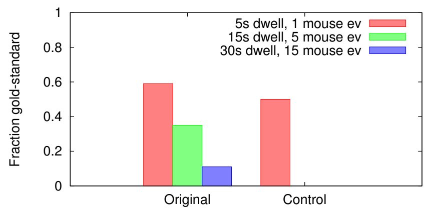
4.5 Validation

Figure 3 compares (normalized) complementary click-spam rates computed by our approach (plotted as error bars) and the (normalized) complementary click-spam rates as reported by Bing and Google for their search ad networks (plotted as bars). We ran another experiment where where we explicitly set our ad campaign to exclude syndicated search partners for one of the search ad networks (plotted as C in Figure 3). The Bing and Google estimates are the ratio between the number of clicks we were charged for (from the billing report), and the total number of landing-page fetches (from our logs). For our approach, we calculate two separate estimates based on the delay and click interstitial pages. The spread of the error bar plots the max and the min of the estimates we compute. The center tick plots the average. The figure plots the estimates for all three ads we flighted. In line with our goals, this figure (and all other figures in this section) are normalized so one of the data points is 1.

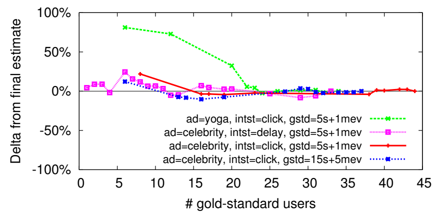
As is evident from Figure 3, our estimate for the yoga and lawnmower ads are in the same ball-park as that reported by Google and Bing. We manually investigated the difference between our estimates and that for the celebrity ad. We found over 50 clicks from sites associated with well-known search redirection viruses where browser toolbars hijack normal user searches and funnel them through affiliate search programs (Section 5.3.1 has more de-



(a) Interstitial page performance



(b) Gold-standard user definition



(c) Convergence

Figure 4: Design space exploration

tails). We were charged for at least 48 of these clicks. Our estimates match the search ad network's estimates recomputed after discounting these clicks. Furthermore, as seen for network C where syndicated search partners are excluded, our estimates closely match that reported by the network. There were no clicks on the control ads on network C, which further supports our high estimate.

Our absolute numbers also agree with public estimates of average click-spam for these networks [3]. Next we drill deeper to validate our design decisions.

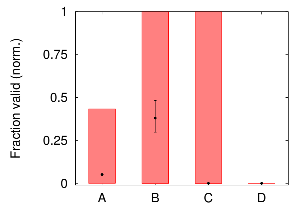
4.5.1 *Interstitial Pages*

Figure 4a plots the fraction of clicks for each interstitial page that reach the landing page for the celebrity ad, and for the corresponding control ad. Note that this fraction drops as the interstitial page changes from clicking a link (29%), to waiting 5 seconds (13%), to solving a CAPTCHA (4%), demonstrating the increasingly higher bar set by the interstitial pages. Interestingly, we find users are more likely to click through to the landing-page than wait 5 seconds. Except for the CAPTCHA interstitial, the fraction reaching the landing page is significantly lower for the control ad than for the original ad; this validates Assumption 1 from the previous section that the interstitial page concentrates non-click-spam traffic (by some unknown amount). Despite the varied interstitial page performance, the estimates computed from the delay and click interstitial converge (in Figure 3) for the experiments where we have a balanced number of converters through the interstitial and direct path, which supports Assumption 2. The CAPTCHA seems to reduce both normal and control traffic to the same low base level regardless of user intent; as a result, it is unsuitable for use in our framework.

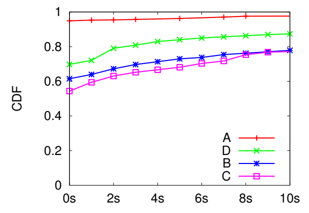
4.5.2 *Gold-Standard Users*

Figure 4b plots the fraction of gold-standard users for the original celebrity ad and the control ad, for three different definitions of gold-standard users. The first definition is, as before, 5s of dwell time and 1 mouse event. The second definition is 15s of dwell time and 5 mouse events. The third definition is 30s of dwell time and 15 mouse events. Note that the fraction of gold-standard users for the control ad is zero for the second and third definition. This validates Assumption 3 that very few users are curious enough to click the control ad. In a real-world setting, we expect advertisers to define gold-standard users based on financial transactions (much tighter than any of our definitions).

We focus next on the sensitivity of the click-spam estimate to the number of gold-standard users. Figure 4c plots the convergence of our click-spam estimate as a function of the number of gold-standard users for various combinations of our ad, interstitial page, and definition of gold-standard user. X-values are driven by the periodicity of ad network reports. Y-values are deltas from our best estimate (last data-point for that series). In each case our estimate converges at or before 25 gold-standard users.



(a) Mobile Ad Network



(b) Dwell Time

Figure 5: Normalized estimates and dwell times for mobile ads

4.6 Search Ad Networks

Figure 3 shows that while reputed search ad networks generally have a good handle on click-spam, a single average click-spam metric across the entire network is of little use due to different keywords experiencing different levels of click-spam. An advertiser cares only about click-spam rates for keywords he is interested in bidding on. The difference in click-spam rates between the celebrity ad and lawnmower ad is up to 20% (normalized).

We omit discussion of 7Search since we did not get any gold-standard users through that network to base our estimates on.

4.7 Mobile Ad Networks

Figure 5a plots our click-spam estimates and the networks' own estimates for the celebrity ad across the four mobile ad networks we measured. Despite running our ads for over a month, and weakening our definition of gold-standard users to only 5s of dwell time (*i.e.*, user spent 5s on our landing page; no tap event required), we failed to attract even five gold-standard users for the yoga and lawnmower ads, and attracted fewer than twenty for the celebrity ad. While, our estimates are below the convergence threshold, to investigate the huge difference in our interim estimates and ad network numbers, we plot the CDF of user dwell-time in Figure 5b.

Ad network A charged us for over a third of the clicks (non-normalized), yet as illustrated by the y-intercept in Figure 5b, over 95% of network A users spent *under a second* on our landing-page! We find evidence of an attack that would result in such a signature in Section 5.4.1. Network D appears to be quite well aware of the poor traffic quality on their network; they charged us for less than 1% of the clicks. Mobile ad network C is a curious case. There is practically no difference between the click-through-rate (CTR) of our original ad and the CTR of our control ad with junk text, suggesting that the content of the ad is irrelevant for users clicking ads on this network. Our Bayesian formula understandably estimates click-spam to be nearly 100% for this network despite the network charging us for most of these clicks.

Our data, although inconclusive, suggests that charges on mobile ad networks do not currently reflect actual user intent.

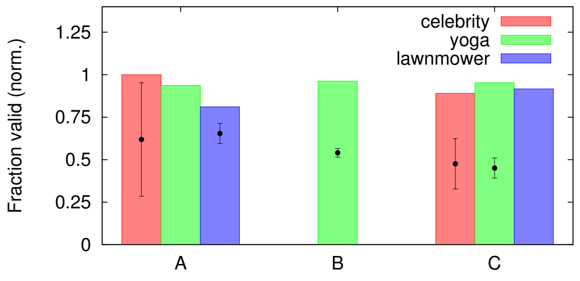


Figure 6: Normalized estimates for contextual and social ads

4.8 Contextual and Social Ad Networks

Figure 6 plots our click-spam estimates, and those reported by contextual and social ad networks we measured. Network B approved only our yoga ad. Click-spam is uniformly higher than that on reputed search ad networks (not apparent due to normalization). The networks do a better job than mobile ad networks in tracking this higher rate of click-spam. Nevertheless, our estimates are uniformly lower than the numbers reported suggesting that these networks do not yet discount all click-spam. Interestingly, network B consistently charged us for more unique clicks than we logged on our server. We speculate network B does not follow the standard practice of suppressing duplicate/double-clicks by a user [39].

5. FINGERPRINTING CLICK-SPAM

Recall in previous sections we assumed that very few people would intentionally click on our control ads (*e.g.*, Figure 2), and substantiated the assumption through the lack of gold-standard users for these ads. Nevertheless, convincing ad networks requires incontrovertible evidence of fraud. One needs to discover the full sequence of events that culminate in the fraudulent click. We manually investigate clicks we receive on control ads. The sophistication and diversity of attacks makes this non-trivial. In this section we describe seven ongoing click-spam attacks we discovered.

5.1 Scale

We were charged approximately $1000 for about 30,000 clicks on all the control ads we created across all ad networks. Our investigations cover 26% of the traffic our control ads attracted on reputed networks; as expected by design, all these clicks were found to be fraudulent in nature. The ad networks typically discounted substantially less than this fraction (between 6–20%). Thus we can confidently claim that some of this fraudulent traffic is currently not caught by ad networks. Note that 26% covers only the traffic we actually investigated; we expect the disparity in discounts vs. fraudulent traffic to grow as we investigate more clicks. That being said, our manual approach is too laborious and not scalable. More automated methods for investigating click-spam are needed.

5.2 Methodology

To prioritize manual investigation of the large number of clicks, we use simple graph-clustering over features in the HTTP request, and detecting heavy-hitting clusters. A naïve approach would be to use the HTTP Referer domain. We found groups of websites on unrelated domains but with nearly-identical layouts (Figure 7), all driving click-spam traffic to our site. This is done presumably to spread out the click-spam through multiple sources in order to operate below detection-thresholds of existing ad networks. Using additional features to cluster such publishers allows us to aggregate them back together and do proper heavy-hitter accounting.

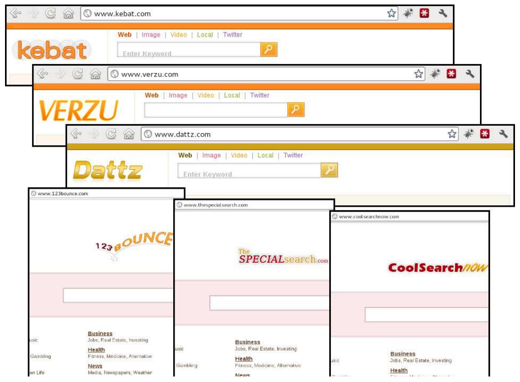
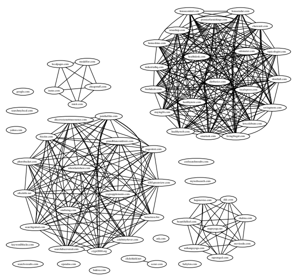
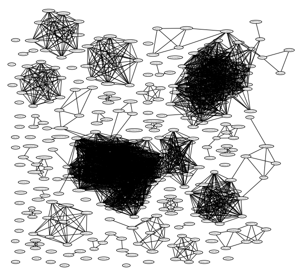


Figure 7: Some Sybil [21] websites driving click-spam to our ads



(a) Clusters (search)



(b) Clusters (mobile)

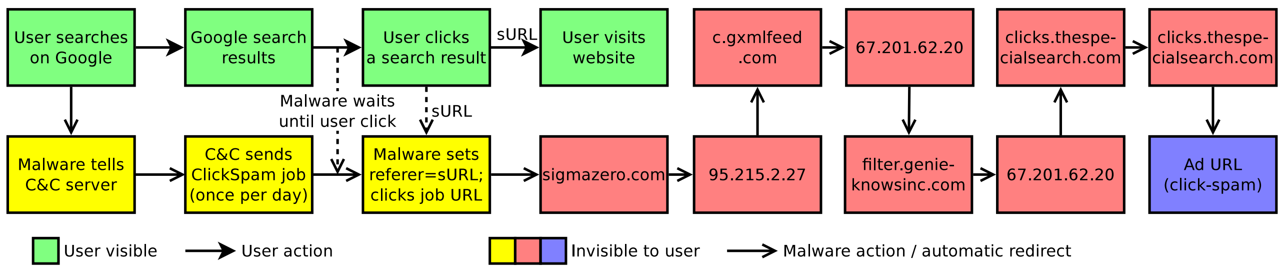
Search Cluster	Signature Type	%
thespecialsearch.com + 2	Malware, Affiliates	5%
scour.com + 3	Badware, Affiliates	14%
Sedo-parked (58+)	Parked Domain	(cloaked)
NS-parked (51+)	Parked Domain	6%
dotellall.com + 20	Arbitrage	18%

(c) Top 5 heavy-hitter cluster for search ads

Figure 8: Graph-clustering and heavy-hitter detection output

Graph-clustering: We induce a graph that spans all publisher domains we see. We do this as follows. For any pair of publishers, we compute a similarity score. We construct a feature vector that consists of various network-level attributes (*e.g.*, Web host IP address, subnet, hosting provider, domain registrar, whois information) as well as HTTP-level attributes in our logs. We assign a weight to each attribute and compute a cosine similarity between the feature vectors of the pair of domains. The similarity score ranges between 1 (identical) and 0 (dissimilar). We add a graph edge between the domains if the similarity score is above some threshold. We find this simple technique is surprisingly robust to our selection of weights and thresholds. In our data distinct cliques emerge at thresholds as low as 0.2 and stay intact beyond 0.9, thus giving us much wiggle room in picking the initial weights and thresholds (which we manually refine iteratively).

Heavy-hitter detection: We use a conductance metric to detect heavy-hitters especially when the clusters do not neatly fall out as distinct cliques. Each node in the graph shares responsibility for clicks originating from another node up to 2-hops away. We found 2-hops to be quite effective since the 1-hop neighborhood was too sparse (due to sparsity in the underlying data), and 3-hop neighborhood resulted in clusters too large for them to represent real-world collusion between bad domains. We compute a badness score for each node as the number of clicks originating in their 2-hop neigh-



User searches on Google
Google search results
User clicks a search result
sURL
User visits website
c.gxmlfeed .com
67.201.62.20
clicks.thespe-cialsearch.com
clicks.thespe-cialsearch.com
Malware waits until user click
sURL
Malware tells C&C server
C&C sends ClickSpam job (once per day)
Malware sets referer=sURL; clicks job URL
sigmazero.com
95.215.2.27
filter.genie-knowsinc.com
67.201.62.20
Ad URL (click-spam)
User visible
User action
Invisible to user
Malware action / automatic redirect

Figure 9: Click-spam through TDL-4 botnet. TDL-4 is stealthy (one click per day), mimics user behavior (click gated on user-click); money flows up affiliate chain (up to 10 redirects seen). Who to blame: sigmazero (chain start) or thespecialsearch (publisher for ad network)?

borhood. We then partition the graph into disjoint clusters by considering nodes in decreasing order of badness as cluster centers, and collapsing nodes within 2-hops from it into its cluster.

We believe better techniques based on learning and mining literature can be designed to find patterns in click-spam data (*e.g.*, [17, 18, 22]). We leave this for future investigations both by us and other researchers. To this end, as mentioned earlier, our raw logs are available online for other researchers to use.

That being said, even our simple technique was able to find meaningful clusters. Figure 8a plots our clustering and heavy-hitter output applied to control ad clicks on Google's and Bing's syndicated search ad networks; all clusters also happen to be cliques. Figure 8c lists the top 5 heavy-hitter clusters. Figure 8b plots the clusters from control ads on mobile ad networks. Next, we dig deeper and discuss some case studies chosen specifically to depict the wide variety and sophistication in current click-spam techniques.

5.3 Click-spam in Search Ads

Click-spam we observed in search ads can be attributed to three main attack vectors: (1) malware and badware, (2) parked domains, and (3) arbitrage.

5.3.1 *Malware and BadWare*

thespecialsearch.com affiliates: We noticed a large number of clicks in our logs that fit the pattern clicks.thespecialsearch .com/xtr_new?q=.... What followed the q= parameter changed from click to click, but almost always was a simple combination of English words (*e.g.*, Team Building or Saving more). 5% of the search clicks in our logs matched this pattern.

Searching online we found malware reports [11] for the Win32/Olmarik (aka TDSS, TDL) botnet that had been observed fetching URLs fitting the above pattern. This particular malware family is incredibly sophisticated [23]. The malware is a generic task execution platform — it contacts its command-and-control server (C&C), downloads an arbitrary task meant specifically for that infection instance, executes it, and repeats the process. The malware hooks into all popular browsers (IE, Firefox, Safari, Chrome), through which it can inject clicks that appear indistinguishable from normal traffic generated by these browsers. The malware can also inject malicious code into iframes the user is browsing, or modify search results before they are shown to the user. It even attempts to cleanse the infected host of *other* malware so it has sole control over the host (and to disrupt other competing botnets).

We found a copy of the malware binary and installed it in a virtual machine. We routed all traffic from the virtual machine through a transparent proxy (running on the VM host) and logged all traffic. We configured the proxy to block SMTP traffic to block malware-generated spam campaigns. We also apply a strict network rate-limit to prevent DoS attacks, and configured our proxy to block requests to the *click URL* of Google, Bing, and other major ad networks to prevent advertisers being charged for clicks made by the malware instance.

How it works: Figure 9 illustrates the process.

1. For every search we performed through the browser, the malware contacted a specific IP address with the following URL: http://63.223.106.16/bV03tDze8...JpdHk=08h. The string of random characters is encoded using base64. It decodes as follows: ver=4.2&bid=noname&aid=50018&sid=0&rd=1307260520&eng=www.bing.com&q=celebrity. Thus, for each search, the malware reported back to its C&C server the version number of the bot, affiliate ID, the search engine where the query was submitted, and the query keywords.

2. In response the C&C server sent back an XML file that directed the bot to effectively click an ad. The XML file encoded the URL to click, the HTTP Referer value to use (*i.e.*, traffic will appear to come from this site), and some accounting information.

It is important to note that the bot did not immediately perform the click after receiving the XML file.

3. The browser showed the (unmodified) search results we had requested as usual.

4. Only when we clicked a search result, the bot kicked into action in the background (*i.e.*, the user doesn't notice anything unusual). The bot contacted the URL it was directed to contact with the appropriate Referer header. This led to a sequence of HTTP and JavaScript redirects (we observed upto 10) that culminated in the final redirection to the click URL for one of the major ad networks. Figure 9 plots the sequence of redirects.

Note: From the user's perspective the malware is completely transparent. The user's search results and subsequent click were not tampered with. The user wasn't shown any extra ads or popups. The user wasn't redirected to an advertiser etc. The malware performed all its activity stealthily in the background.

5. The bot then ceased this behavior for 24 hours (as long as the IP address stayed the same). The following day the bot would repeat this activity, perform one click, and become dormant again.

When we acquired a new external IP address (easy to do since the ISP uses DHCP), the bot came out of dormancy, performed one click, and resumed dormancy.

Discussion: It is clear that the C&C server is tracking which bots are active and from where, and ensuring that across the botnet each IP address is used only for one click in a 24 hour period — an extremely low threshold that would likely not raise any flags. Furthermore, when the bot does click, it is gated by a legitimate user click (on the search page), which would defeat click-spam detection mechanisms that look for deviations from normal user behavior (*e.g.*, [35]). Lastly, since it hooks into a regular browser, and forges the referer of a legitimate site, click-spam detection techniques that look for deviation at the HTTP or HTML layer would fail.

Who made money: The penultimate website (thespecialsearch-.com) made money from the ad network. Note this website also shows up in our list of Sybils (Figure 7). The long chain of redirects to thespecialsearch.com presumably identify the chain of affiliates, each making some fraction of the money the next one in the chain did. We noted that the malware used different affiliate chains based on geography (*i.e.*, it used one when in the US, and a different one when in another country). This suggests that the same malware is performing click-fraud for different "customers". Thus the botmaster controlling the botnet likely made money as well.

scour.com affiliates: A large number of clicks were through clicks.-scour.com. Scour is a meta-search engine that aggregates results from Google, Yahoo, Bing into a single search result page. It pays users to search through it, and to vote or comment on individual results. It has an affiliate program where registered affiliates are paid for users they refer to Scour.

How it works: We found a browser toolbar that hijacks the user's searches through Scour (but doesn't automatically click on ads.) The affiliate ID (6678) is hardcoded in the scour.com URL the toolbar points the browser to. The same affiliate ID shows up in our logs. The toolbar, which many anti-virus companies classify as the Scour redirect virus, is extremely hard to remove [9]. Additional search redirect viruses (unrelated to Scour) that we see clicks from include search-results.com, mywebsearch.com, search.babylon.com, search.alot.com and search.conduit.com. These sites explain the discrepancy between our estimate in Figure 3 and the ad networks'.

Who made money: As before, the publisher (scour.com) made money from the ad network, and the affiliate presumably made some fraction of that from Scour.

5.3.2 Parked Domains

sedo.com parkers: While investigating a set of about 35 clicks from a particular domain registered by Sedo, a domain registrar, we stumbled across 57 other domains also hosted by Sedo and in our various logs. All these domains are parked domains. A parked domain is a domain name that is registered, but not in use. The registrar typically points DNS for that domain name to a Web server that serves up a "This site is under construction" or similar message, followed by a set of ads that the user may or may not click.

For these specific parked domains, however, Sedo would automatically redirect the browser to the ad click URL.

How it works:

1. A domain name registered by Sedo was either never used, or its previous owner vacated it before the registration expired. Sedo serves the parked page to users reaching the domain name.

2. A user may reach the parked domain name through many ways. He may have mistyped a domain (*e.g.*nsdi.com is a Sedo parked domain that appeared in our logs; other examples include blogdpot.com a mistyping of blogspot.com, and icicbank.com a typo for icicibank.com)

In other cases where the owner was using the domain but is no longer, links to the domain when it was active may have been posted on forums, exchanged in emails, indexed by search engines etc., and users may click on these links in the present. There are also reports of adult link-exchange networks that launder traffic through parked domains [13].

3. The user is usually shown an ad-laden parking page.

4. If, however, the parked domain is one of these 58 domains (possibly more) and the user is geolocated to a certain set of countries, which include UK, Brazil, Italy, India, China, Spain and Argentina (but notably, *not* the US), Sedo serves up a piece of JavaScript that in effect automatically clicks the first ad link without ever serving up the parking page.

5. The automatic click initiates a chain of redirects (we observed upto 4) many of which (roughly one in three) culminate in a redirect to clicks.scour.com (but lacks an affiliate ID). Scour then shows ads from major ad network.

Based on the referrer we see in our logs (which appears to be a search query on Scour) we found that the domain of the Sedo parked page is linked to the search query on Scour. For example, the publisher URL for ad clicks originating at icicbank.com is ....scour.com?q=icic+bank. Indeed this is how we discovered the set of 58 domains that auto-redirect users. For each Scour query in our logs we attempted to guess the Sedo parked domain by appending common top-level domains (.com, .net, etc.), and checking which were parked, and then determined which countries they auto-redirect for using PlanetLab nodes located in 45 countries.

6. The remaining auto-redirects reach either thespecialsearch.com — encountered earlier in the context of malware, or searchmirror.com. Both these sites further auto-redirect to the ad click-URL.

The ultimate ad URL in these cases is encoded in each of the intermediate redirects starting from the very first redirect initiated at the Sedo parked domain. Thus the decision for which ad to click was made right at the onset.

7. In many cases, the first ad is for the correct version of the mistyped domain (*e.g.*, the ad posted by icicibank.com is the top ad on icicbank.com). Thus when the user is automatically redirected, he may never realize that he mistyped the domain, although under the covers icicibank.com (as an advertiser) has had to pay for a user that was about to reach his Web page anyway, and several parties made money in the process.

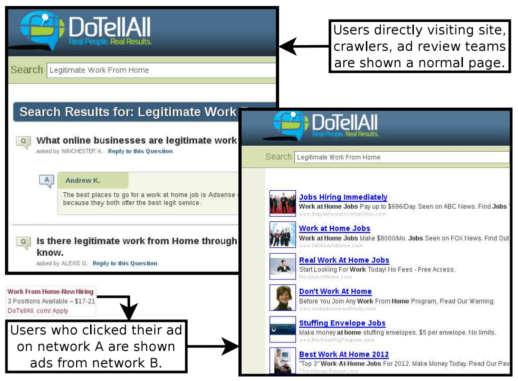
Discussion: Since Sedo parked domains redirect via a chain of affiliates, detecting Sedo (or the Sedo customer) as the root-cause requires reverse-engineering the chain (in this case through a query parameter on scour.com, which taken out of context appears to be a normal search query). More deviously, the user (*e.g.*, that typed icicbank.com) would engage normally with icicibank.com since he anyway meant to type the latter. Thus any advertiser driven engagement metrics would appear perfectly normal. Discovering such patterns automatically is likely to be highly challenging, but would illuminate a fraction of click-spam that is virtually undetectable. Detection is only part of the problem however.

Sedo is benefiting from a ad network policy that does not forbid its mode of operation. Parked domains are not only allowed to show ads, ad networks expose special APIs to help them in doing so [7]. Worse, even though ad networks have mechanisms to allow advertisers to block certain classes of traffic (*e.g.*, traffic through proxy-servers), ad networks *do not* allow advertisers to block traffic from parked domains.

Who made money: thespecialsearch.com and scour.com made money from major ad networks, some fraction of which, as before, traveled through the affiliate chain to the Sedo parked page.

networksolutions.com: NetworkSolutions, another domain registrar, has a similar model as Sedo, but does not automatically redirect. They account for 6% of the clicks we see for control ads.

In one scenario we found that even though the owner of www.-noblenet.org (a library website) is actively using it, NetworkSolutions is showing a parked page for noblenet.org that, at first glance, appears to be a library page except all links are ads that direct the


Users directly visiting site, crawlers, ad review teams are shown a normal page.
Users who clicked their ad on network A are shown ads from network B.
DoTellAll
Real People. Real Results.
Search

Legitimate Work From Home

Search Results for: Legitimate Work

What online businesses are legitimate work
asked by WINCHESTER A. Reply to this Question

Andrew K.
The best places to go for a work at home job is Adsense
because they both offer the best legit service.

Is there legitimate work from Home through
know.
asked by ALEXIS G. Reply to this Question

Work From Home-Now Hiring
3 Positions Available – $17-21
DoTellAll. com/ Apply

Jobs Hiring Immediately
Work at Home Jobs Pay up to $696/Day. Seen on ABC News. Find Jobs
www.StayatHomeIncomeMom.com

Work at Home Jobs
Work at Home Jobs Make $8000/Mo. Jobs Seen on FOX News. Find Out
www.SBWorkAtHome.com

Real Work At Home Jobs
Start Looking For Work Today! No Fees - Free Access.
NextJobAtHome.com

Don't Work At Home
Before You Join Any Work From Home Program, Read Our Warning.
www.workathome-authority.com

Stuffing Envelope Jobs
Make money at home stuffing envelopes. $5 per envelope. No limits.
www.BestMailingProgram.com

Best Work At Home 2012
"Top 3" Work-At-Home Jobs For 2012. Make Money Today. Read Our Rev
The-Money-Report.com

Figure 10: Arbitraging click-spam traffic through a fake site

user away from their intended URL. Note that here the user did not make a typo; he simply omitted the www, which is often acceptable.

Discussion: As before, this is largely a policy issue. Major ad network policies for parked domain affiliates states that they must not violate trademarks and copyrights [1]. NetworkSolutions does reserve for themselves the right to served parked pages for a domain (or sub-domain) in its terms-of-service (TOS) that customers must agree to. It is unclear whether benefiting from someone else's domain constitutes copyright or trademark infringement, and if it does, whether it can be overridden by the TOS. This is a loop-hole NetworkSolutions benefits from.

Who made money: NetworkSolutions made money from major ad networks if the user clicked a link on the parked domain.

5.3.3 Advertising Arbitrage

dotellall.com family: We next focus on the cluster of dotellall.com and 20 other related domains that account for 18% of the traffic for our search control ads. The entire cluster of websites on the surface appear to be lively social question answer forums (users ask questions, and post answers), but when we posted questions and answers on one of the sites, it disappeared after a few days, and the site was restored to its pristine condition. We noticed that over time the questions and answers do not change. No question has the date/time when it was asked or answered. For one of the sites, we found the content was blatantly copied from other locations on the web. As best as we can tell, the entire family of sites is an incredibly elaborate (and realistic) sham.

It was extremely puzzling as to how they attract traffic. Clearly users wouldn't frequent a fake social site. We couldn't find links to malware. The sites weren't typos of other popular sites (although one is named livingfrugal.com, which is similar to the popular livingsocial.com). Confusing us further, we (initially) couldn't find ads on their pages. It took us a long time (and a considerable amount of serendipity) to determine how this family of sites makes money.

How it works:

1. We serendipitously discovered that the family of sites *advertises* heavily on search and contextual ad networks. It advertises to the tune of thousands of ads, for a wide spread of (long-tail) keywords. As a result, they show up on low-popularity searches or low-quality publishers. On low-quality publishers ad links are almost indistinguishable from content. Being the only ad for many low-popularity searches, their ad is often placed above search results. Whenever there is competition, however, their ad is typically ranked much lower (*e.g.*, 7th or 8th position in the sidebar).

This suggests that they likely bid mere pennies for these thousands of ads, but nevertheless manage to acquire long-tail traffic.

2. When a user clicks one of these ads, he is taken to the site. The site *shows ads only when the user arrives through an ad-click* (identified through a URL parameter in the landing-page URL the site registered for its ads in the previous step). Figure 10 shows screenshots of the site for users arriving either directly or through an ad-click.

This second set of ads is from a different major search ad network. Based on the keywords highlighted, we believe this second set of ads are more expensive. The site filter-ins these higher-value ads by stuffing keywords into the ad request.

Thus the family of sites acts as an advertiser with one search/contextual ad network, and as a publisher with another search ad network.

3. Note that the users reaching the site have already displayed a propensity to click ads. Presented with low-quality content, and prominent ads on the top, it is likely a large fraction of these users click the ad on the site. When they do, the second ad network pays the site some fraction of the ad revenue from the lucrative ad, say a dollar, while the site likely pays the first ad network mere cents for getting the user (and pockets the difference).

Discussion: Arbitrage has been long known to be an issue in ad networks [24]. However, such elaborate fake sites can be incredibly hard for a human at an ad network to detect (given limited time to investigate publishers). Recall how the site does not even show ads if navigated to directly. Discovery is only half the problem.

The second half is that these sites are not violating ad network policy. An advertiser may show ads on the landing-page. A publisher may advertise his site. A publisher may provide useful content hints in the ad request. A poor quality page and a prominent ad box is bad user-experience, and an SEO optimized publisher ultimately costs the advertiser, but does not violate current policy.

Who made money: The dotellall.com family of sites likely made a lot of money from one search ad network, for inexpensive traffic it bought from the other ad network.

5.4 Click-spam in Mobile Ads

We next turn our attention to mobile ads, which as we found in Section 4, are challenging even for reputable ad networks to detect click-spam in. Figure 8b pictorially shows why. First, because mobile advertising is a relatively new market, large legitimate content providers have not yet replaced fly-by-night operators that exist to make a quick buck. Indeed many mobile sites on which our ads were shown serve primarily adult content; the abundance of these sites mirrors the state of the web two decades ago when banner ads first started appearing on similar sites. The two large cluster are different adult entertainment networks, one hosted in Turkey, and one in Denmark. We do not investigate these clusters.

5.4.1 Mobile Games

Ant-smasher and similar games: At least 2% of clicks on control ads came from smartphone games that all require the user to tap the screen close to where the ad is displayed. One such example is the Ant-smasher iPhone app where ants randomly walk around the screen up to (and under) where the ad is shown in the game, and the user must tap the ant before it disappears from the screen to progress in the game. We installed the games directing the most traffic and confirmed the following modus operandi.

How it works:

1. A mobile game developer accidentally (or intentionally) places the in-app advertising control close to where the user must tap, or drag things to, in order to succeed in the game.

2. Given the tiny screen real-estate, the user is prone to mistapping. When he does so, the browser navigates to the ad-click URL.

3. The user may realize his error and switch back to the game. The browser, which in the mean time has already begun fetching the ad landing-page, aborts the attempt. As a result, the user will appear to have spent very little time on the advertiser's page. We saw exactly this behavior on our mobile ads — 95% of users spent less than a second as mentioned earlier.

Discussion: The core issue here is the advertiser being charged despite the user not spending any time on the landing page. It is hard for an ad network to know how long the user spent on the advertiser's site. If it relied on the advertiser to get this information, the advertiser could easily lie to get a discount. Solving this without modifying the browser, and without hurting the user experience is a non-trivial problem.

One mitigating approach would be to audit games and apps that trick users into mistapping on the ad. Doing so would likely spark an arms race for apps intentionally exploiting this loop-hole, but would at least protect advertisers from apps accidentally triggering this. Unfortunately, ad networks are making it harder for advertisers and independent third-parties to identify bad apps. During the course of our study, one major mobile advertising network stopped sending the application ID in the HTTP Referer.

Who made money: The app made money from the ad network.

5.4.2 WAP Phones

waptrick.com and other sites: There is a sizable number of WAP phones (phones with a limited browser that access the web via a WAP proxy) that mobile ads are shown to. Nearly 42.1% of traffic on our mobile control ads are from these sources. We loaded a number of implicated sites with our browser's user-agent set to that of a WAP browser.

How it works:

1. Sites that cater to WAP users as well as WAP proxies optimize the page content for display on feature phones, potentially stripping away icons, colors, or sidebar content that visually differentiates an ad from a normal link.

2. Combined with the extremely small screen, and clunky (keypad-based) navigation on non-smart phones, the user clicks a link either unintentionally, or without knowing that it is an ad.

Discussion: Proxies are the biggest hurdle in tracking down bad WAP sites that confuse the user. As mentioned, less than 36% of our clicks had the HTTP Referer we need to track it back to the originating website and confirm that it intermixed ads with content. While one might wish for legacy phones to die out, it is unlikely to do so in developing countries in the near future. Advertisers wishing to reach a global market will have to contend with click-spam originating through these vectors.

Who made money: The WAP website made money from the ad network. For websites that have arrangements with proxies, the proxy operator potentially made some fraction of that money.

5.5 Epilogue and Future Work

Investigating 26% of clicks on our control ads, we find the five classes of invalid clicks discussed above. We believe there are more classes of dubious traffic lurking in our data, and are investigating more automated means of reconstructing the attacks. In any event, we find that click-spam is by no means a solved problem.

We also find that while there is a policy component to many of the case-studies we presented, there is also an associated technology (and research) component to proactively discover attacks.

Mobile is a particularly tricky case where much of the telemetry needed for detecting click-spam doesn't exist. Given the large role mobile advertising is expected to play in the coming future, research in this space is both important and timely.

6. RELATED WORK

Related work falls into three distinct categories.

Measuring Traffic Quality: There is surprisingly little past work in systematically measuring the quality of click traffic. [26] develops a learning algorithm for estimating the true CTR of an ad in the presence of click-spam. [41] measures traffic from bulk traffic providers and finds some providers to be qualitatively worse than ad networks. Startups including Adometry, Visual IQ and ClearSaleing that claim to be able to estimate click-spam rates provide no transparency into the specifics of their methods; furthermore, these approaches apply only at the granularity of entire ad networks, which we found is insufficient information for advertisers. Our click-spam estimation approach, which is grounded in our Bayesian framework and validated through extensive measurements, is the first principled approach an advertiser can independently apply at the granularity of his individual ads.

Documenting Click-Spam: The second category of related work is a snapshot-in-time of click-spam attacks, much like the case-studies presented in this paper. Daswani et. al. [19] give a good introduction to online advertising, pricing models, and online advertising fraud. Botnets like Clickbot.A [20], TDL-4 [36] and other botnets [34] have been used for click fraud. More recent work describes fraud in ad exchanges [38]. Individual advertisers, and security researchers have documented many more attacks in blog posts and white-papers [8,10,13]. Each of these has been an ad-hoc targeted investigation given a specific publisher or attack vector. Our generic clustering and heavy-hitter detection approach instead starts from raw click logs to automatically identify (and prioritize) potential publishers/attack vectors for targeted investigations.

Mitigating Click-Spam: The third category of related work aims to identify individual clicks as click-spam so they can be discounted. Bluff Ads [25], on which we base our control ad design, are ads with unrelated targeting information (*e.g.*, dog food ads for cat lovers). Clicks on Bluff ads are assumed to be click-spam, which the ad network should discount. While we subscribe to this assumption, we differ in how such ads should be used. [25] suggests blacklisting users that have above-threshold clicks on bluff ads. There are two problems. First, this only applies to click-spam driven by malware. In the non-malware scenarios we discovered, blacklisting the user serves little purpose since the bad publishers get a steady stream of unwitting users (false-negatives for Bluff ads); furthermore, the legitimate clicks of blacklisted users on good publishers would also get discounted (false-positives). The second problem is that even for click-spam driven by malware, it wouldn't work. The malware we analyzed performs one click per day. If Bluff Ads were to be shown 1% of the time, it would take on the order of a 100 days to blacklist a user. The cost to the ad network would be 1% of their revenue (hundreds of millions of dollars for reputed networks), which would be unacceptably high. We use control ads in a different way; we use it sparingly to collect data ($1000 represents a negligible fraction of ad network revenue), from which we then extract click-spam signatures that apply more broadly.

Other approaches to mitigating click-spam include SbotMiner [40], Sleuth [33] and Detectives [32]. SbotMiner tries to identify bot activity by using KL-divergence to detect change in query distributions, followed by pruning of false positives due to flash crowds, by leveraging heterogenity for genuine users. Sleuth uncovers single publisher fraud by finding correlation in multi-dimensional data; however, they claim that the technique is suitable only when the botnet uses tens of hundreds of IP addresses. Detectives detects coalition hit inflation attacks by their similarity seeker algorithm; it discovers coalitions made by pairs of fraudsters, which is then enhanced in [31] by finding groups of fraudsters. All these approaches apply only to botnet and malware driven click-spam, which is dwarfed by other sources of click-spam in our data.

Premium Clicks [27], access control gadgets (ACG) [37] and CDN fraud prevention [30] focus on mitigation strategies that go beyond botnets. Premium clicks employs economic disincentives that devalue clicks from non-gold-standard users. ACGs ensure authentic UI interactions by users clicking a link. CDN fraud prevention proposes a heavy-weight challenge-response protocol for publisher-payee CDN models. While the first assumes an alternate ad economy, the second and third (applied to ad networks) require re-architecting the browser, or the ad network infrastructure. None of these approaches apply to click-spam in existing ad networks.

Focusing squarely on existing ad networks, Camelot [28] is Google's click-fraud penetration system. It can test the susceptibility of the network to known click-spam signatures, but does not itself detect new signatures. [39] describes the invalid click detection system inside Google, without identifying the specific heuristics that are used to identify invalid clicks. No heuristic is perfect. Our data shows click-spam is still an open problem despite these deployed systems.

7. CONCLUSION

In this paper, we take a systematic look at click-spam. We propose the first methodology for advertisers to independently measure click-spam rates on their ads. We also develop an automated methodology for ad networks to proactively fingerprint different simultaneous click-spam attacks. We validate both methodologies using data from major ad networks. We then conduct a large-scale measurement study of click-spam across ten major ad networks and four types of ads. In the process, we identify and perform in-depth analysis on seven ongoing click-spam attacks not currently caught by major ad networks. We conclude that even for the *largest* ad networks, click-spam is a serious problem, and is especially rampant in the mobile advertising context. Given the evolving nature of click-spam, we believe that click-spam is an open problem that requires a concerted effort from the research community to tackle. To this end we have publicly released the data gathered for this paper to aid other researchers in the design of novel click-spam defense techniques.

Acknowledgments

We'd like to thank Jigar Mody, Matt Graham, our shepherd Kirill Levchenko, and our anonymous reviewers. This paper is much improved thanks to their valuable feedback and suggestions.

8. REFERENCES

[1] AdSense for domains program policies. http://support.google.com/adsense/bin/answer.py?answer=96332.

[2] The adsense revenue share. http://adsense.blogspot.com/2010/05/adsense-revenue-share.html.

[3] Click Fraud Falls in Q4 2010. http://searchenginewatch.com/article/2050117/Click-Fraud-Falls-in-Q4-2010.

[4] Click fraud rampant in online ads, says bing. http://www.theaustralian.com.au/media/click-fraud-rampant-in-online-ads-says-bing/story-e6frg996-1226056349034.

[5] Cloaking and Faking the Referrer. http://kbeezie.com/view/cloaking-and-faking-referrer/.

[6] For Impatient Web Users, an Eye Blink Is Just Too Long to Wait. http://www.nytimes.com/2012/03/01/technology/impatient-web-users-flee-slow-loading-sites.html.

[7] Google AdSense for Domains. http://www.google.com/domainpark/index.html.

[8] Google Click Fraud Inflates Conversion Rates and Tricks Advertisers into Overpaying. http://www.benedelman.org/news/011210-1.html.

[9] Google Redirect Virus: How to Remove. http://www.pcmag.com/article2/0,2817,2370676,00.asp.

[10] International Cyber Ring That Infected Millions of Computers Dismantled. http://www.fbi.gov/news/stories/2011/november/malware_110911.

[11] Malware connection report. http://www.malware-control.com/statics-pages/03aaf7c8e47ef32e8de23dfe9215d4a5.php.

[12] Stealing Clicks. http://www.forbes.com/2007/09/21/google-click-forensics-tech-secure-cx_ag_0924fraud.html.

[13] Uncovering an advertising fraud scheme. Or "the Internet is for porn". http://www.behind-the-enemy-lines.com/2011/03/uncovering-advertising-fraud-scheme.html.

[14] Upcoming changes in Google's HTTP Referrer. http://googlewebmastercentral.blogspot.com/2012/03/upcoming-changes-in-googles-http.html.

[15] L. V. Ahn, M. Blum, N. J. Hopper, and J. Langford. Captcha: using hard ai problems for security. EUROCRYPT'03, 2003.

[16] Click Quality Team, Google Inc. How Fictitious Clicks Occur in Third-Party Click Fraud Audit Reports. http://www.google.com/adwords/ReportonThird-PartyClickFraudAuditing.pdf.

[17] G. Cormode and S. Muthukrishnan. What's hot and what's not: Tracking most frequent items dynamically. In *Proceedings of ACM PODC*, July 2003.

[18] G. Cormode and S. Muthukrishnan. Improved data stream summaries: The count-min sketch and its applications. *Journal of Algorithms*, 2004.

[19] N. Daswani, C. Mysen, V. Rao, S. Weis, K. Gharachorloo, and S. Ghosemajumder. *Crimeware:Understanding New Attacks and Defenses*, chapter Online Advertising Fraud. 2008.

[20] N. Daswani and M. Stoppelman. The anatomy of clickbot.A. In *Proceedings of HotBots*, 2007.

[21] J. R. Douceur. The Sybil Attack. In *Proceedings of IPTPS '02*.

[22] C. Estan and G. Varghese. New Directions in Traffic Measurement and Accounting. In *Proceedings of ACM SIGCOMM*, Aug. 2002.

[23] A. M. Eugene Rodionov. The evolution of tdl: Conquering x64. http://go.eset.com/us/resources/white-papers/The_Evolution_of_TDL.pdf.

[24] B. Geddes. Search arbitrage: Web blight or brilliant marketing strategy? http://searchengineland.com/search-arbitrage-web-blight-or-brilliant-marketing-strategy-10768.

[25] H. Haddadi. Fighting online click-fraud using bluff ads. *SIGCOMM Comput. Commun. Rev.*, 2010.

[26] N. Immorlica, K. Jain, M. Mahdian, and K. Talwar. Click Fraud Resistant Methods for Learning Click-Through Rates. In *Proceedings of the Workshop on Internet and Network Economics (WINE '05)*.

[27] A. Juels, S. Stamm, and M. Jakobsson. Combating click fraud via premium clicks. In *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, 2007.

[28] C. Kintana, D. Turner, J.-Y. Pan, A. Metwally, N. Daswani, E. Chin, and A. Bortz. The goals and challenges of click fraud penetration testing systems. International Symposium on Software Reliability Engineering, 2009.

[29] H. Lieberman. Letizia: An agent that assists web browsing. In *International Joint Conference on Artificial Intelligence*, 1995.

[30] S. Majumdar, D. Kulkarni, and C. V. Ravishankar. Addressing click fraud in content delivery systems. In *In Proceedings of the 26th IEEE INFOCOM International Conference on Computer Communications*, 2007.

[31] A. Metwally, D. Agrawal, A. El Abbad, and Q. Zheng. On hit inflation techniques and detection in streams of web advertising networks. ICDCS '07, 2007.

[32] A. Metwally, D. Agrawal, and A. El Abbadi. Detectives: detecting coalition hit inflation attacks in advertising networks streams. WWW '07, 2007.

[33] A. Metwally, F. Emekçi, D. Agrawal, and A. El Abbadi. Sleuth: Single-publisher attack detection using correlation hunting. *VLDB*, 2008.

[34] B. Miller, P. Pearce, C. Grier, C. Kreibich, and V. Paxson. What's clicking what? techniques and innovations of today's clickbots. In *Proceedings of the 8th international conference on Detection of intrusions and malware, and vulnerability assessment*, DIMVA'11, 2011.

[35] B. Mordkovich and E. Mordkovich. Click fraud and how to counteract it in ad campaigns. In *Pay-Per-Click Search Engine Marketing Handbook*, 2005.

[36] E. Rodionov and A. Matrosov. The evolution of TDL: Conquering x64. Technical report, 2011.

[37] F. Roesner, T. Kohno, A. Moshchuk, B. Parno, H. Wang, and C. Cowan. User-driven access control Rethinking permission granting in modern operating systems. IEEE Symposium on Security and Privacy, 2012.

[38] B. Stone-Gross, R. Stevens, A. Zarras, R. Kemmerer, C. Kruegel, and G. Vigna. Understanding fraudulent activities in online ad exchanges. Internet Measurement Conference, 2011.

[39] A. Tuzhilin. The lane's gift v. google report. http://googleblog.blogspot.in/pdf/Tuzhilin_Report.pdf.

[40] F. Yu, Y. Xie, and Q. Ke. Sbotminer: large scale search bot detection. WSDM '10, 2010.

[41] Q. Zhang, T. Ristenpart, S. Savage, and G. M. Voelker. Got traffic?: an evaluation of click traffic providers. In *Proceedings of the 2011 Joint WICOW/AIRWeb Workshop on Web Quality*, WebQuality '11, 2011.