Characterizing Large-Scale Click Fraud in ZeroAccess

Paul Pearce[†◇] Vacha Dave[□] Chris Grier[†◇] Kirill Levchenko[§] Saikat Guha[¶]
Damon McCoy[‡] Vern Paxson[†◇] Stefan Savage[§] Geoffrey M. Voelker[§]

[†]University of California, Berkeley [◇]International Computer Science Institute
[□]Microsoft [§]University of California, San Diego
[¶]Microsoft Research India [‡]George Mason University

ABSTRACT

Click fraud is a scam that hits a criminal sweet spot by both tapping into the vast wealth of online advertising and exploiting that ecosystem's complex structure to obfuscate the flow of money to its perpetrators. In this work, we illuminate the intricate nature of this activity through the lens of ZeroAccess—one of the largest click fraud botnets in operation. Using a broad range of data sources, including peer-to-peer measurements, command-and-control telemetry, and contemporaneous click data from one of the top ad networks, we construct a view into the scale and complexity of modern click fraud operations. By leveraging the dynamics associated with Microsoft's attempted takedown of ZeroAccess in December 2013, we employ this coordinated view to identify "ad units" whose traffic (and hence revenue) primarily derived from ZeroAccess. While it proves highly challenging to extrapolate from our direct observations to a truly global view, by anchoring our analysis in the data for these ad units we estimate that the botnet's fraudulent activities plausibly induced advertising losses on the order of $100,000 per day.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Invasive Software*; K.4.2 [**Computers and Society**]: Social Issues—*Computer Abuse and Crime*

Keywords

Click Fraud; Malware; ZeroAccess; Cybercrime; Measurement

1. INTRODUCTION

Profit drives modern cybercrime. Investments in malware, botnets, bullet-proof hosting, domain names, and other infrastructure must all be justified by the greater revenue brought in by the scams that use them. Thus, scammers relentlessly innovate to identify more lucrative niches to maximize their returns (e.g., counterfeit goods, fake anti-virus, ransomware, credit card theft, cryptocurrency mining, etc.). Monetization, however, also presents some of the greatest risks, since it represents both a single point of vulnerability in the business model and a potential means of forensic attribution [18, 24, 27]. Thus, the ideal monetization strategy for an Internet scammer would not only tap into great wealth, but would also effectively launder the money trail as well. As we will describe in this paper, the modern advertising ecosystem is remarkably well-suited to this need, and modern click fraud provides perhaps the *non plus ultra* of all such monetization schemes.

First and foremost, online advertising represents an enormous revenue stream, generating over $20 billion in revenue in the first half of 2013 and growing at an estimated 20% per year [35]. Moreover, by design, online advertising is a low-friction market designed to engage the broadest possible set of participants, and thus presents few barriers-to-entry for potential bad actors. Unsurprisingly, criminal groups have developed a range of techniques for generating synthetic advertisement clicks for profit at the expense of legitimate advertisers and ad networks [4]. Indeed, such *click fraud* accounts for as much as 10% of all advertising clicks, potentially defrauding advertisers of hundreds of millions of dollars annually, with some experts predicting the rate increasing by more than 50% per year [38].

As a further difficulty, the ecosystem of online advertising is highly complex, and while occasionally traffic flows directly from publisher to advertiser, in the common case instead a vast array of middlemen—syndicators, subsyndicators, traffic aggregators and resellers—separates the two endpoints of each ad click. While the path of such a click on the Internet is nominally visible (a chain of redirects from one domain to the next), the chain of payment remains largely opaque. Those on the buy-side and the sell-side of the traffic ecosystem negotiate their contracts bilaterally, with a wide range of terms and conditions. Thus, no single party comes remotely close to having comprehensive visibility of who gets paid what for any given click. Finally, as we describe, modern click fraud platforms can engage with a wide range of ad networks, who in turn *mix* this traffic with other, more legitimate sources, further complicating any forensic analysis.

In this paper we illuminate these problems by characterizing the click fraud perpetrated by *ZeroAccess*. Active in a variety of forms since 2009 [9], ZeroAccess was one of the largest botnets in operation, commanding an estimated 1.9 million infected computers as of August 2013 [33]. More importantly, ZeroAccess was particu-

CCS'14, November 3–7, 2014, Scottsdale, Arizona, USA.

ACM 978-1-4503-2957-6/14/11 ...$15.00.
http://dx.doi.org/10.1145/2660267.2660369 .

larly known for monetization primarily via click fraud (with losses to advertisers estimated at $2.7 million per month [40]). However, while the technical aspects of ZeroAccess's design and operations (e.g., infection vector, peer-to-peer C&C protocol) are well documented [9, 33, 40, 41], the nature of its click fraud behavior and the attendant monetization has seen less study. In part, this is because such analyses require a range of disparate vantage points, including of the botnet itself, of infected hosts, and of impacted publishers.

By combining an array of data sources, including peer-to-peer measurements, C&C telemetry from botnet infiltration, and click information from one of the top ad networks, we have constructed a deep analysis to illuminate the rich, intertwined nature of modern click fraud and the advertising ecosystem it exploits. In particular, our work makes three contributions: First, we provide a detailed description of the click fraud component of ZeroAccess, the innovations it introduced in hijacking high-quality user search traffic, and the side effects by which we were able to track its activity. Second, we show how to match botnet membership data, network telemetry, and ad network click streams using a combination of timing information and reactions to external events (in this case the Microsoft-initiated takedown of ZeroAccess click fraud infrastructure and the botmaster's immediate responses). Finally, using this technique we identify with high confidence 54 individual "ad units" (here roughly corresponding to distinct traffic sellers) whose traffic volume (and hence revenue) was predominantly rooted in ZeroAccess. By anchoring our analysis in these ad units, we roughly estimate that the botnet produced on the order of a million fraudulent clicks a day, plausibly inducing advertising losses on the order of $100,000 per day. However, the uncertainties involved in extrapolating to this global picture loom large enough that we must caution that this reflects a coarse-grained estimate, and accordingly we discuss the challenges involved in forensic accounting of click fraud payouts.

Taken together, our work illustrates the complex nature of the click fraud problem and highlights the need for much better mechanisms for correlating traffic and payment streams.

2. BACKGROUND

As background, we provide an overview of the advertising ecosystem on the Web, how attackers defraud Web advertising networks, specifically the two methods by which ZeroAccess has perpetrated click fraud at scale, and insight into the attempted takedown of the botnet by Microsoft. Throughout we highlight related work that has also studied the ZeroAccess botnet and various aspects of click fraud.

2.1 Web Advertising and Click Fraud

When a user issues a search query, the resulting page includes organic search results, for which the linked Web sites do not pay the search engine, as well as paid search ads, for which the linked Web sites (the advertisers) pay the search engine for inclusion. These ads are typically placed above the organic results or alongside on the right. They are formatted similarly as search results except for a darker shade background and the word "Ad" or "Sponsored" somewhere nearby.

Search engines select ads based on the user's search query. The search query is distilled into a group of keywords after normalizing to remove misspellings and resolve ambiguities. Advertisers indicate keywords (or groups of keywords) for which they would like their ad to be considered for inclusion in search results.

Search ad syndication network. Search engines partner with thousands of Web sites, services, and applications—collectively called publishers—to extend the reach of the search engine's advertisers to users beyond the search engine. Publishers include blog sites, news sites, niche search engines, ad-supported browser addons, and other ad networks.

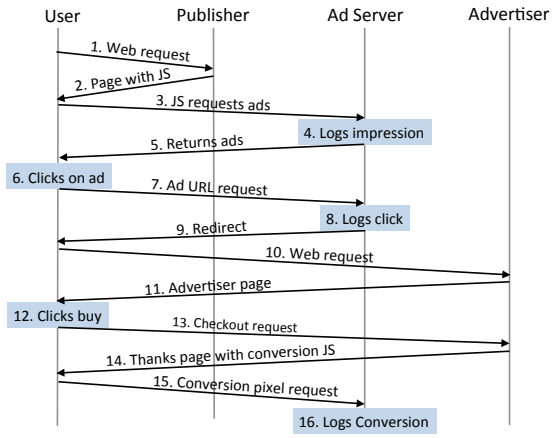

Figure 1: Anatomy of a typical ad click, showing the various HTTP requests associated with a user clicking on an advertisement, leading them to an advertiser's *landing page*, and from there possibly to additional interactions.

The publisher sends the user's search query from the publishers Web site or app, or in the case of blogs and news sites the context of the article the user is reading, to the search engine's ad server to retrieve relevant ads in exchange for a cut of the advertising revenue. The publisher can display relevant ads by embedding JavaScript provided by the ad server, which directs the user's browser to fetch the ads as illustrated in steps 1–6 in Figure 1; alternatively, the publisher can directly fetch relevant ads through server-to-server communication between the publisher and the ad server (not shown in the figure). If the user clicks the ad, the user is taken to the advertiser's site through a series of redirects as shown in steps 6–11 in the figure. Each interaction at the ad server is logged along with the publisher identifier for the publisher responsible for the traffic.

Publishers can, in turn, (sub)syndicate to other downstream publishers according to the search ad network's policy. The downstream publisher requests ads from the intermediate publisher, which then fetches them from the ad server. The search engine pays the intermediate publisher a cut of the ad revenue. The intermediate publisher pays the downstream publisher a smaller cut and retains the rest in exchange for the service it provided. Thus the search engine does not directly deal with the downstream publisher, and in many cases never learns of its existence. Subsyndication arrangements exist to help search ad networks scale to hundreds of thousands of publishers by distributing the management overhead down tiers of aggregators.

For example, Google and Bing syndicate search ads to other search engines including InfoSpace and Yahoo respectively [12, 31]. InfoSpace and Yahoo show these ads on Web sites owned and operated by them, but additionally subsyndicate to smaller networks like Publishers Clearing House and Chitika respectively, which then further subsyndicate to yet smaller publishers.

Revenue share. Search ads are typically charged on a cost-per-click (CPC) model, i.e., the advertiser pays only if the ad is clicked.[1] For clicks on ads shown by a syndicate publisher, the

[1]CPC ads predominate today in terms of ad revenue [35]. While other charging models exist for other types of advertising, e.g., cost-per-impression (CPM) for display ad networks, and cost-per-acquisition (CPA) for affiliate ad networks, these other models are not relevant to this paper.

search engine typically retains 30% of the amount charged to the advertiser, and pays a 70% cut to the publisher [21]. Publishers that sub-syndicate set their own revenue sharing agreements with their downstream publishers.

Click fraud and blending. Revenue sharing with (sub)syndicate publishers creates an economic incentive for these publishers to fraudulently attract clicks on ads shown by them. While a well-known publisher would lose reputation if caught engaging in click-fraud, followed by financial losses if ejected from the syndication network, this disincentive does not exist for less well-known publishers. Less reputed publishers may not have a reputation to protect, or may be able to reattach to the syndication network at a different point if ejected. As a result, click fraud from relatively unknown publishers is rampant [38].

Click fraud techniques have evolved considerably in the past several years. The direct approach of hiring people to click on ads (termed click farms) [10], or running stand-alone scripts that repeatedly retrieve the URLs associated with ads to simulating user clicks (stand-alone bots) [26], are now easily detected by an ad network's defenses. More complex approaches include search engine hijacking [36], where a malicious in-browser plugin replaces ad links in results returned for user searches by other ads, and the rise of click fraud botnets like ZeroAccess that coordinate large numbers of malware-infected hosts to fetch and click ads unbeknown to the user.

A number of case studies have chronicled the evolution of click fraud botnets over the years, ranging from the early Clickbot.A botnet [3] to the TDL-4 botnet [36], the Fiesta and 7cy botnets [32], and ZeroAccess itself [40, 34]. Security researchers have similarly documented more attacks in blog posts and in whitepapers [7, 8, 19]. Unlike previous studies, however, we analyze the ZeroAccess botnet primarily from the perspective of the advertising network, supplemented with insight from operational data derived from its P2P infrastructure and use of DNS. As a result, we are able to present the first comprehensive characterization of monetization, distribution, and activity of a massive click fraud botnet.

At the same time, malicious (sub)syndicated publishers have become better at avoiding detection or identification. Using techniques such as referrer cloaking [6], or fetching ads through other publishers that use the direct server-to-server mechanism, many sub-syndication publishers remain completely anonymous. Intermediate publishers complicit in this activity blend in traffic from non-malicious small publishers to present a cleaner appearance in the aggregate to their syndication parent. This blending results in even reputed publishers unwittingly facilitating click fraud.

Click fraud mitigation and smart-pricing. Due to the large number and relative anonymity of publishers in the (sub)syndicate network, search engines rely primarily on automated means, supplemented with limited manual investigations, to protect their advertisers from click fraud. Rule-based techniques [39, 22], correlation analysis [29, 30, 42], and bluff ads [16] have focused on detecting clicks from infected users before the advertiser is charged. Clustering [4] and anomaly detection [5] have focused on detecting malicious publishers.

Smart-pricing [11] takes an economic approach to mitigating the impact of click fraud. At a high level, smart-pricing maintains a normalization factor between 0 and 1 for each publisher based, in part, on the probability of *conversions* generated by traffic sent by that publisher [13]. Conversions are actionable business results as defined by the advertiser such as an online sale, newsletter sign-up, etc. Advertisers inform the ad server of a conversion by embedding JavaScript provided by the ad server on the Web page corresponding to the conversion (Figure 1, steps 12–16). The ad server uses cookies to link the conversion to an earlier ad click, which is then associated with the publisher that was responsible for the click. For a simplistic illustration, consider otherwise identical publishers X and Y that both send 100 legitimate users, but X additionally blends in 100 fraudulent clicks. Consider further that 10 legitimate users convert in each case. X therefore has half the conversion probability of Y (5% vs. 10%). Smart-pricing normalizes CPC for X to effectively be half of that for Y, such that the advertiser has the same effective cost per conversion.[2] Implicit in the smart-pricing mechanism is the assumption that click fraud traffic will not result in conversions, an assumption that the Serpent module (Section 2.2) of ZeroAccess specifically tries to defeat.

Our ZeroAccess case study informs this debate with the latest generation of click fraud botnets, and identifies potentially fruitful directions for building the next generation of mitigation capabilities.

2.2 Evolution of ZeroAccess

ZeroAccess is a vast and complex peer-to-peer (P2P) botnet that serves as a delivery platform to distribute a variety of malware modules, each with its unique command-and-control (C&C) and monetization strategy [1].

Initial reports of the ZeroAccess rootkit date to 2009 [9]. By 2010, the main module delivered via ZeroAccess was FakeAV, which claimed to be anti-virus software to extort users into paying money to remove fictitious infections. An estimated 250,000 computers were infected. A radically new version of ZeroAccess emerged in May 2011 that changed its distribution, communication, and monetization strategies [33]. This version spread via exploit packs (e.g., BlackHole [17]) and social engineering [15, 41].

The defining feature of this iteration of the botnet was the introduction of a decentralized, P2P communication protocol. The protocol used cryptography and obfuscation as well as other common P2P features such as "supernodes" that served to orchestrate large portions of the network's activity. The network allowed the botmasters to maintain decentralized control while relaying commands and payloads to infected computers worldwide [40].

In addition, the monetization strategy changed with this generation. ZeroAccess moved away from FakeAV payloads and instead began distributing Bitcoin miners and click fraud modules.[3] From a technical perspective, the primary click fraud malware module used in this era operated in the indiscriminate "auto-clicking" fashion we describe in more detail later in this section.

This iteration of the botnet also saw an increase in botnet population. At the height of infections in early 2012, estimates placed the botnet population at over 500,000 [28].

In July 2012, ZeroAccess evolved into the form predominant as of its disruption by Microsoft in December 2013. According to Symantec, by August 2013 this generation had an estimated population of over 1.9 million computers [33]. This iteration again included several changes to the malware structure, the protocol, and the payloads. The monetization strategy also evolved, introducing and massively disseminating a new click fraud payload performing search-hijacking, which we have named Serpent and discuss later in this section. Serpent has been linked to the MagicTraffic click fraud affiliate program [20].

[2] In practice, smart-pricing takes multiple features into account, and applying the normalization factor given dynamic bidding and publisher diversity is not as straight-forward

[3] ZeroAccess's shift away from FakeAV occurred just before a major takedown that resulted in the closure of most FakeAV programs [23].

P2P communication. The P2P C&C substrate of ZeroAccess functions solely to deliver modules to infected machines. This P2P protocol is well understood, has common P2P properties such as peer lists and supernodes [33, 37, 40], and allows ZeroAccess infected computers to communicate with each other directly without the need for a centralized C&C server. Without payloads, ZeroAccess infections simply maintain their P2P membership and functionality, but perform no malicious actions on their own.

Infected ZeroAccess machines generate UDP P2P traffic on four well-known ports. The protocol is constructed in such a way that super nodes in the network have a vantage point of a large portion of the infected population. ZeroAccess has distributed several modules with various monetization strategies over its five-year lifespan. In this paper we focus on the two most recent modules distributed throughout 2013. Both these modules perform click fraud and both have a separate HTTP-based C&C channel distinct from the P2P network.

Auto-clicking module. The ZeroAccess auto-clicking module performs click fraud by simulating normal Web browser behavior of a user clicking an ad. These clicks occur rapidly, require no user participation, and are not visible to the user of the infected computer. Some users, however, may be alerted to the presence of this module by the increased network activity; in one instance, we observed about 50 MB of click traffic per hour.

The auto-clicking module invokes an actual client Web browser, enabling realistic Web browsing behavior including proper handling of HTML, CSS, JavaScript as well as browser-specific quirks. The module periodically contacts the auto-clicking C&C to fetch a list of publisher Web sites for the bot to visit. It navigates to the URL provided by the C&C in a hidden window, locates the ad on the page, and simulates a user clicking on the ad by requesting the advertiser's URL. The browser faithfully follows the sequence of redirects from the URL until it loads the advertiser site. The auto-clicking module then closes the hidden window and starts anew with the next publisher in the list provided by the C&C. Notably, the auto-clicking module does not simulate any user actions on the advertiser page and thus does not trigger any conversions.

Serpent module. The Serpent module interposes on a user's normal interaction with various search engines to redirect the unwitting user to an ad. Whereas the auto-clicking module simulates a user, Serpent sends a real user to the advertiser.

The Serpent module includes a browser component that lays dormant until the user performs a search on one of many search engines. The module silently sends a copy of the search query to the Serpent C&C. The C&C responds with a set of URLs that correspond to ad click URLs for ads related to the search query. Meanwhile, the browser renders the original search result page as the user expects. When the user clicks on a search result Serpent intercepts the click and prevents the browser from navigating to the site intended by the user. Instead, the module redirects the browser to one of the URLs it received from the C&C, in effect creating the appearance of the user having clicked an ad after performing a search on the publisher's site (as opposed to the site on which the user actually performed the search). If the user does not notice having been unwittingly redirected to the advertiser's site (a different site than intended), the user may continue browsing as normal. Since the advertiser site is relevant to the user's search query, the user may even convert. Note that since all Serpent activity is gated on legitimate user activity, this type of click fraud attack is extremely hard for an ad network to detect.

Parallel to this search-hijacking activity, Serpent also used a separate C&C protocol to perform a type of auto-clicking. This auto-clicking protocol and ad network structure were separate from the original auto-clicking module, although the auto-clicking behavior is largely similar.

DNS queries. Serpent ships with a list of C&C IP addresses hard-coded into each module. These C&C IP's are encoded inside ASCII strings that appear to be domain names ending in `.com`, but are in fact an obfuscated encoding of the numeric IP address. For each Serpent C&C function the malware selects one of these pseudo-domains, decodes the domain into an IP address, and then initiates a connection to that IP. Note that the malware does not need to perform a DNS query since the IP address is encoded in the domain name itself. However, for some unknown reason, the malware authors coded the module to perform a DNS query via Google's public DNS server each time it decodes the pseudo-domain—resulting in a DNS query for each C&C operation—and discards the resulting DNS response (if any).

Most of the ZA Serpent pseudo-domains were not registered prior to our study. For this study we registered all available pseudo-domains, and by observing these DNS queries we are able infer operations the malware is performing. Section 3 discusses this DNS data, and the inferences it enables, in more detail.

Takedown attempt. On December 5, 2013, Microsoft's Digital Crimes Unit and Europol orchestrated a takedown of the Serpent and auto-clicking C&C servers [25]. Since both click fraud modules had a centralized C&C server, a simple seizure of these machines was able to temporarily disrupt click fraud activity. Since the ZeroAccess P2P substrate was still intact the perpetrators were able to distribute an updated module with new C&C IP addresses within hours and fraud resumed. For reasons unknown, the following day the malware authors distributed a new set of modules that halted all click fraud activity but left the P2P network intact. Inspection of these modules revealed the ASCII text "WHITE FLAG" in apparent surrender [14].

Three months later, on March 21, 2014, the malware authors used the P2P network to revive the botnet by issuing a new Serpent module. This module removed all search-hijacking functionality, leaving only the auto-clicking aspect of the Serpent module intact. As of August 2014, the ZeroAccess Serpent module is active and performing auto-clicking click fraud.

3. DATA SOURCES AND QUALITY

Our study draws upon extensive, disparate sets of data. The individual datasets all suffer from limitations or skews of various forms. Constructing our large-scale picture in a sound fashion frequently requires cross-correlating different data sources in order to filter out spurious activity and bring out the underlying signals that reflect different facets of ZeroAccess's click fraud activity. In this section we sketch each of the data sources, our use of it, and its associated data-quality issues. A summary of all data used in this study is given in Table 1.

Click data. Via our partnership with one of the top ad networks, we acquired access to the "clicks" that the network logged from Nov 28–Dec 6 2013, spanning the Takedown event. The ad network views this data as highly sensitive from a business perspective and thus in our study we use the data at reduced fidelity and present certain facets of it only in relative terms rather than using absolute values.

In abstract terms, each "click" datum consists of a count of ad-following Web requests observed arriving at the ad network from a given source, during a given interval, and associated with a given ad unit. In the context of this work, an ad unit maps roughly to distinct traffic sellers. In particular, our click data aggregated individual clicks into tuples consisting of ⟨ one-hour interval, /24 subnet, ad unit, count ⟩. We believe this data to be of high quality: none of our

Dataset	Granularity	Quantity
ZA DNS telemetry, Dec 1–Dec 4		
Timestamp	millisecond	16,208,758
Domain	Full query	12
IP	/24	336,609
Supernode data, Dec 1–Dec 6		
Timestamp	millisecond	260,811,204
IP	/32	1,137,118
IP	/24	637,736
Bot type	32/64 bit OS	2
ZA module distribution, Jun 18–Apr 25		
Timestamp	second	51
Module ID	64-bit ID	51
Module MD5	Full MD5 sum	51
Milker data, Sept 10–Dec 5		
Ad replacements	Full URL	1,766
Redirects	Full URL	10,796
Click data, Nov 28–Dec 6*		
Timestamp	hour buckets	Over 10TB of raw ad server logs
IP address	/24	
Clicks	hourly sum	
Ad unit	Anon ID	
Ad unit data, Nov 28–Dec 6*		
Timestamp	hour buckets	Around 2TB of raw ad server logs
IP address	/24	
Clicks	hourly sum	
Ad unit	Anon ID	

Table 1: Summary of datasets used in our study. *Precise quantities for click and ad unit data intentionally omitted due to business sensitivities. We only use this data in aggregate.

analyses or cross-checks raised questions regarding any potential inaccuracies or missing values.

Ad unit data. Our ad network partner also provided information for selected ad units in terms of the conversion percentage for their ads, and their mean and median smart-pricing discounts, with these latter being in normalized form so as not to reveal sensitive business information. The data included those ad units we identified as very likely tainted by ZeroAccess activity, as well as two randomly sampled populations of comparable size, and global baseline figures aggregated across all ad units.

This data covers the same time period as the click data discussed above. It allows us to explore the relative effects of ZeroAccess's activity compared to regular ad unit costs and conversion efficacy.

ZeroAccess DNS telemetry. As discussed previously, vestigial code in Serpent's modules leads it to issue DNS requests to a number of different domains based on its current activity. Each different Serpent function has a different set of domains associated with these lookups; as far as our extensive analysis could tell, the malware chooses randomly among the set for a given function. The malware does not process any replies it receives for associated DNS requests, and thus does not even require that the domains exist.

Indeed, during our study most of the domains did not exist. Beginning on Nov 28 2013 we registered all such ZeroAccess domains not already registered (6 out of 12), and immediately began receiving queries for them. The queries all came from Google's public DNS server, 8.8.8.8, and thus nominally did not identify the associated ZeroAccess system. However, Google includes support for an EDNS0 option [2] that identifies the subnet originally associated with requests that resolvers such as theirs issue. Thus, our data from this source has the form of tuples consisting of ⟨ timestamp, domain, /24 subnet ⟩, where the timestamp has high precision (sub-second) as recorded at our DNS server.

The domains we registered included 3 of the 5 associated with Serpent modules reporting that users had searched, and the sole domain associated with Serpent reporting that a user had clicked on a substituted search ad. (The other domains related to functionality not relevant for our click fraud study.)

At first blush this data held promise for illuminating the fine-grained activity of (nearly) each Serpent infectee. However, extensive analysis of the data revealed that lookups did not have a one-for-one correspondence to individual Serpent actions. The data also included significant activity clearly associated with timers, but not so sharply timer-driven that we could readily distinguish it from legitimate activity. However, the data *does* provide us with a virtually complete list of IP addresses associated with ZeroAccess's Serpent module, which allowed us to cross-check against ZeroAccess supernode data to assess its completeness.

Supernode data. From a partner we acquired a list of nodes discovered by crawling the ZeroAccess peer-to-peer network from Dec 1–6 2013. Each entry consists of ⟨ timestamp, address, ZA-network ⟩, where timestamp is high precision, address is the full /32 IP address, and ZA-network reflects on which of the two ("32-bit" and "64-bit") P2P networks the crawl found the node.

Note that these nodes should reflect a *superset* of Serpent nodes, since not all ZeroAccess infectees ran Serpent. Thus, from a click fraud perspective this data is potentially more complete than that derived from the DNS data described above. However, by cross-checking the /24s seen in the DNS data with the equivalent /24s seen in this data, we found that the supernode data included only 200,708 of the 336,609 Serpent /24s. This discrepancy is explained by the design of the P2P crawl resulting in a incomplete view of the P2P network. Thus we conclude that this data reflects only about 60% of the entire ZeroAccess population. This shortfall becomes crucial in our subsequent analysis as we aim to determine which ad units present in the ad network's click data clearly had significant ZeroAccess-generated clicks in their traffic.

ZA module distribution information. Beginning on Dec 4 2013, we ran ZeroAccess infectees in 56 long-running, contained VM environments to allow them to participate in the P2P network and thus receive module updates. Whenever one of them received a new module, we detected the event in real-time and captured a copy of the module. This data allows us to track the evolution of the botnet's functionality.

In particular, this data source allows us to track the botnet's partial recovery post-Takedown (when the auto-clicking modules were updated), which we use in our subsequent analysis as one of the signals for identifying ad units whose traffic has significant taint from ZeroAccess activity. It also allows us to study the blending of Serpent traffic with auto-clicking.

Milker data. Drawing upon extensive reverse engineering, we developed an emulator for the ZA-C&C protocol used by Serpent to request ads to substitute into those present in a user's search results. The emulator enabled us to "milk" ad replacements out of the C&C server by repeatedly requesting ads from it, though the C&C server appeared to "dry up" in its ability to provide new replacements after repeated queries. (The rate at which this drought occurred varied.)

Each C&C reply provided a URL to click on (along with a matching Referer). We followed the URL using a fully functional headless Web browser displaying the `curl` User Agent (such a User Agent prevents advertisers from being charged from our seeming clicks), which in general would continue for each click until it ends at an advertiser's landing page. In total, we captured the redirection chains for 1,766 such clicks for a small sampling of search terms we selected from trending shopping queries. 367 of the clicks transited our partner ad network.

Based on our DNS telemetry correlated with our click data, and supported in part by our small scale milking experiment, we believe

that the global impact of ZA was likely an order of magnitude larger than seen by our partner ad network.

4. ANALYZING FRAUD

To build up our overall picture of ZeroAccess's large-scale click fraud activity, we start with the data most central to assessing ZeroAccess's impact, namely the Click data provided by our ad network partner. We then draw upon our other data sources to identify activity (primarily ad units) associated with ZeroAccess clicks. We employ two main approaches—analyzing the behavior of sources to the Takedown event, and looking at source "demographics" in terms of which subnets contribute clicks—and then cross-correlate these two to develop our overall picture.

4.1 Takedown Dynamics

The Dec 5 2013 Takedown event abruptly severed C&C for ZeroAccess's click fraud activity, which in principle should manifest as a striking change in the activity of any source fueled by ZeroAccess clicks. We then face the basic question of how to reliably detect this presumably sharp signal without inadvertently treating benign variations in traffic rates as stemming from ZeroAccess.

Figure 2 shows the relative clicks per hour for four exemplary ad units, with the leftmost vertical line marking Takedown. Here we have partitioned the examples into "good" ad units that do not primarily receive ZeroAccess traffic and "dirty" ones that do (using a methodology we will describe shortly). Normal click behavior prior to Takedown exhibits the expected diurnal pattern. For good ad units, this pattern continues (Figure 2a), while for dirty ones, precisely at Takedown their clicks cease or dramatically decrease (Figures 2c and 2d).

However, we cannot simply attribute ZA-dirtiness to *any* source that precipitously fell at the time of Takedown, because such behavior also manifests for benign sources (per Figure 2b). Such behavior can be attributed to advertising budget depletion or the end of specific advertising campaigns. This behavior necessitates the need for incorporating multiple signals to soundly identify dirty sources.

One such signal concerns the effects of a secondary Takedown event. The dotted line in Figure 2 corresponds to the release of a new auto-clicking module as a response to the attempted takedown. This module contained new C&C IP addresses, and resulted in auto-clicking click fraud resuming for a subset of the botnet. We can see that some sources exhibit a spike in activity coincident with this module's appearance (Fig. 2d) while others do not (Fig. 2c).

Armed with these signals, we attempted to robustly identify dirty sources based on statistical testing. We undertook numerous evaluations looking for robust indications of behavioral shifts. For example, we compared the volume of each source's click activity as seen during the hour of the Takedown (denoted H-hour) versus during the previous hour ($H-1$). Using the null hypothesis that the relationship between these counts remained unaffected by the Takedown, we applied Fisher's Exact Test to assess the consistency of the shift between those hours as seen on a non-Takedown day versus that seen on Takedown day.

The test identified a large number of sources with statistically significant deviations in the shift for those hours, even for quite low p values (e.g., 0.001).[4] Manual inspection of the most extreme examples confirmed that many appeared to clearly reflect instances of ZeroAccess-affected behavior.

However, when we then tested one non-Takedown day against *another* non-Takedown day, we likewise found many statistically significant deviations, which clearly had nothing to do with the Takedown and thus presumably nothing to do with ZeroAccess activity. The clear conclusion (backed up by manual assessments of exemplars) is that the null hypothesis often fails to hold due to frequent *non-stationarity* in the data. That is, a given click source's patterns from one day to the next can exhibit striking variations; two separate days are not well-modeled as independent samples from the same underlying population. (Figure 2b shows such an instance.)

This lack of stationarity significantly complicates our analysis, and means that statistical testing can only serve as a guide to help direct manual analysis due to the risk of false positives. (In addition, the non-stationarity serves as a caution for applying any sort of training-based machine learning to the problem of identifying fraudulent ad click sources.)

4.2 Subnet overlap

Conceptually separate from the Takedown dynamics, we can seek to identify dirty ad units by assessing each source's degree of "ZA taint" (i.e., proportion of individual sources potentially associated with ZeroAccess activity). This taint can then provide us additional context with which to interpret a given source's Takedown dynamics.

ZA taint. For each ad unit, we consider its full set of clicks. Due to restrictions of the dataset, we identify clicks based on one-hour granularity and /24 subnet of the source of each click. For each hour, we compute the proportion of subnets appearing in the ad unit's traffic that also appeared during that hour in our Supernode data. We then term the mean value of that proportion as the ad unit's "ZA taint". (We limit this computation to hours up to but not including H-hour, so as to not skew the taint by the Takedown dynamics.)

Limitations of the supernode comparison. The Supernode data suffers from incompleteness. By the nature of the ZeroAccess P2P network, no single node has a complete vantage point of the entire network, and thus the matching for our taint computation may incur significant false negatives. To gauge the impact of these false negatives, we compared the ZeroAccess supernode data with our DNS telemetry. Our DNS telemetry gives us complete /24 subnet views of the Serpent portion of the ZeroAccess botnet, but no vantage of the auto-clicking portion. Still, any Serpent subnet missing from the Supernode data likely reflects incompleteness in the latter. If we look for DNS telemetry subnets to show up within 1 hour in the Supernode data, then we find about an 80% match. If we look for matches within 1 minute, this drops to 60%. Infectees repeatedly show up in the Supernode data with such frequency that this latter comparison may in fact provide a better estimate than the former (which allows for a degree of IP address churn to introduce false positives).

A further limitation of our data is the reduction of address information to /24 subnets, which results in us tainting all clicks from a given subnet as bad. Such false positives will result in some benign ad units having increased ZeroAccess taint.

4.3 Combining Signals

Given the limitations of the Supernode data, we cannot use the presence of these IPs as a sole indicator. Instead, we look for ad units that exhibit both a high fraction of ZeroAccess taint as well as uncharacteristic behavior at the various Takedown-related events.

To combine notions of taint and Takedown response, we calculate the ratio of the amount of click traffic at H-hour to the previous hour, $\frac{H}{H-1}$. Ratios closer to 0 denote sharp drops in traffic. Dirty ad units might not exhibit a ratio of exactly 0 because of traffic from

[4] We used a one-sided test since we only had interest in a shift towards an abnormally low H-hour level.

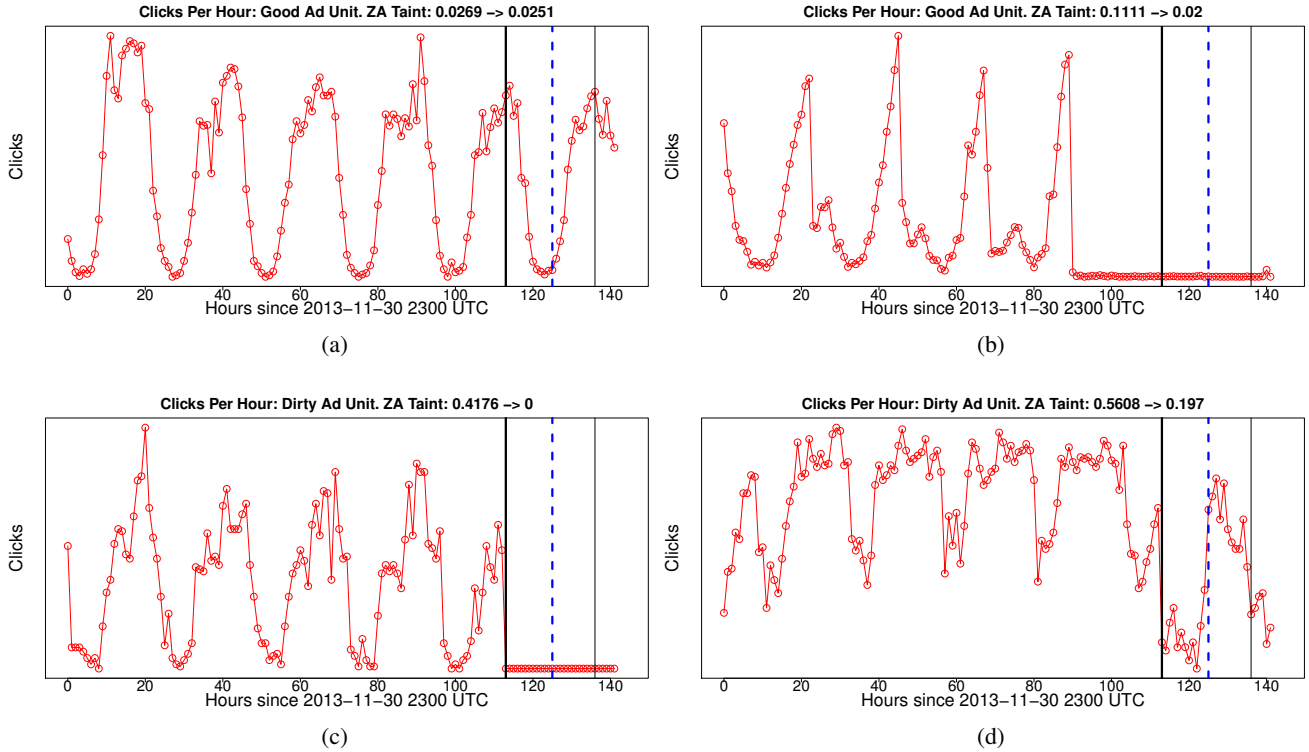

Figure 2: Clicks per hour prior to and after Takedown for 4 exemplary ad units. The thick vertical line marks Takedown (8AM PST, Dec 5 2013). The dotted line to its right indicates the release by the ZeroAccess botmaster of a new auto-clicking module to counteract the Takedown. The thinner line to its right indicates the "WHITE FLAG" module release (per Section 2.2). The plot titles also give the "ZA taint" (Section 4.2) pre- and post-Takedown (per Section 4.2). (a) shows a typical large ad unit that does not see a significant drop in click traffic post-Takedown. The ad unit in (b) exhibits a large drop in traffic that occurred *prior* to Takedown, highlighting the surprising benign dyanmics manifest in the data. (c) shows an ad unit whose traffic almost entirely consisted of fraudulent clicks. The ad unit in (d) clearly had a substantial proportion of ZeroAccess traffic, but mixed in with legitimate or non-ZeroAccess clicks.

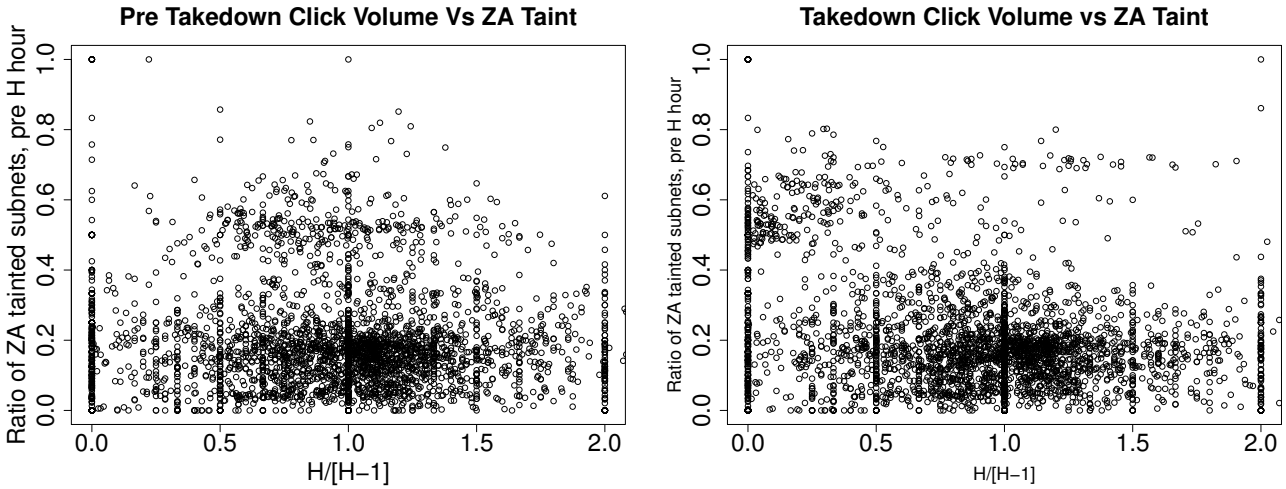

Figure 3: Comparison of ad-unit click volume before and after the Takedown hour (H-hour) to the amount ZeroAccess taint. The left shows this comparison on the day prior to Takedown, and the right across Takedown. A large population of ad units has both higher-than-average ZeroAccess taint and also shows a strong shift towards much fewer clicks on Takedown. The harmonics at $x = 0.5$, $x = 1.0$, $x = 1.5$, etc., arise from ad units with very low click volumes.

other sources, or click-fraud events already "enqueued" from prior to the Takedown.

Figure 3 shows a comparison of the traffic-drop ratio to ZeroAccess taint for each ad unit. We do this for H-hour on both the day before Takedown (left) and the day of Takedown (right). The first plot shows a population of ad units with higher-than-average ZeroAccess taint centered around a drop ratio of 1 (no major change across H-hour) on the day prior to Takedown. The second plot shows a dramatic shift to a greatly reduced volume of traffic at H for that same population of ad units with high ZA taint, with a large number of them approaching zero traffic.

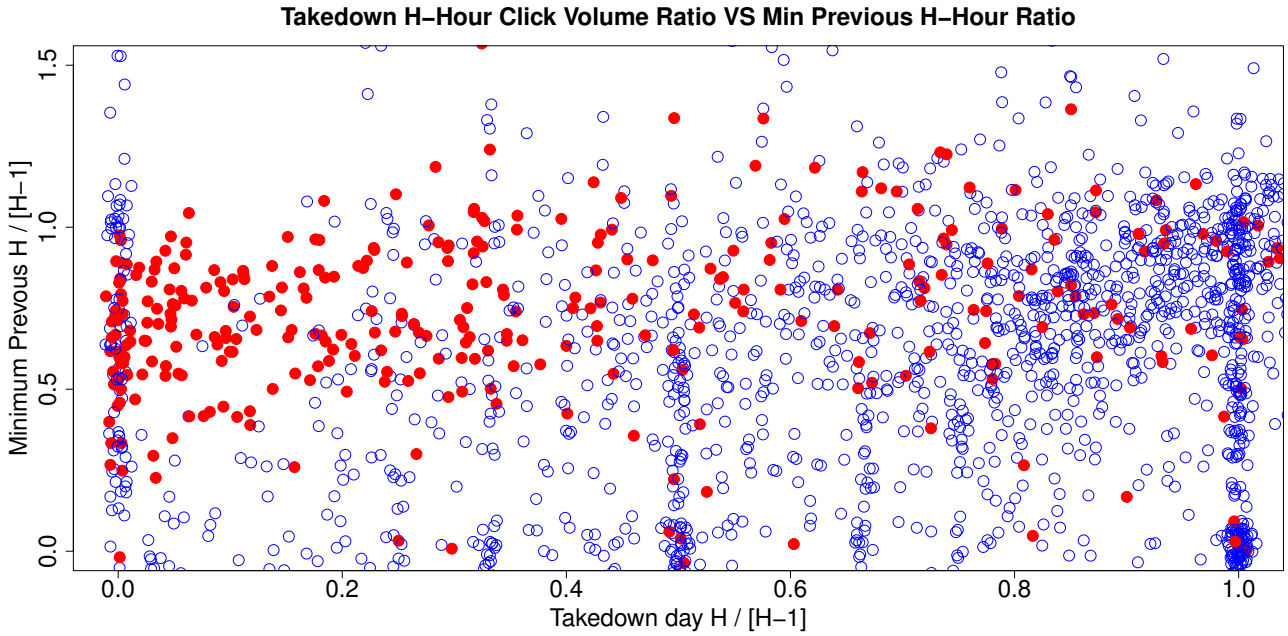

Figure 4: Comparison of the stability of traffic drops across H-hour per ad unit. We plot the Takedown H-hour ratio against the minimum H-hour ratio seen on any other day. We denote ad units with ZA taint ≥ 0.4 using solid red circles, and others with hollow blue circles. Note that we have added small amounts of Gaussian jitter to prevent coincident points from completely overlapping.

Figure 4 explores the stability of traffic of each ad unit across H-hour over time. We plot the Takedown traffic ratio across H-hour against the minimum H-hour ratio for all previous days. Solid red circles denote ad units with ≥ 0.4 ZA taint. A large population of ad units with high ZeroAccess taint exhibits atypically low H-hour ratios (below 0.5) compared to their previous minimum (above 0.5). We deem these ad units as likely ZeroAccess dirty.

Secondary takedown information. Figure 5 shows the behavior of each ad unit across another Takedown event, the distribution of the auto-clicking module shortly after Takedown, which we denote as R-hour. Not all ZeroAccess infectees ran the new module, but those that did would result in a dramatic rise in click activity at R-hour compared to just before R-hour; a change in the opposite direction as what we would observe at H-hour. The figure shows the traffic drop ratio across R-hour plotted against the drop across H-hour, again using solid red circles to denote ad units with $\geq$ 0.4 ZA taint. Clearly, many ZeroAccess-tainted ad units showed sharp reactions to both Takedown and the advent of the new module.

The distribution of an updated auto-clicking module at R-hour immediately restarted ZeroAccess's click fraud. We would expect the resumption of click fraud to result in dirty ad units having low ZeroAccess taint in the time between H-hour and R-hour (since no C&C servers exist to drive click fraud), followed by a sudden increase in taint at R-hour.

Figure 6 compares taint as seen during these two regions of time (H-hour to R-hour, and after R-hour). The left plot reflects a (presumably typical) non-Takedown day, while the right plot shows Takedown day. The shift between the two is clear: a large number of ad units that had significant taint prior to Takedown (solid red) show a jump in pre-R vs. post-R taint, reflecting the activation of a significant number of ZeroAccess subnets within their traffic.

Finalizing determination of dirty ad units. Using these signals we generated a large set of potentially dirty ad units, about 2,000 in number. We then manually inspected the activity of each and produced a list of 54 we deem with high confidence as significantly "dirty" with ZeroAccess traffic.

5. ASSESSING ZA-DIRTY AD UNITS

Having identified these dirty ad units, we now employ them as the basis for evaluating the strategy used by the more sophisticated Serpent module to circumvent smart-pricing. We also leverage this conservative set of ad units to estimate the total amount of fraud perpetrated by ZeroAccess, an undertaking that strongly emphasizes the need for better attribution in the ad ecosystem.

5.1 Conversions and Smart-pricing

We next use our data to explore potential reasons for why ZeroAccess may have used both auto-clicking and Serpent styles of click fraud. Recall from Section 2 that ad networks use automated approaches to either detect (and block) specific click fraud attacks, or mitigate the impact of click fraud (e.g., through smart-pricing) when ad networks can only estimate the amount of click fraud in the aggregate. Clearly Serpent, which confuses a real user into clicking, is harder to detect (and block) than auto-clicking.

We first examine whether we see evidence that Serpent indeed increases the chance of users converting and, if that is the case, whether the blending of Serpent and auto-clicking avoids the smart-pricing mitigation.

Figure 7 shows a clear but modest correlation between ZeroAccess's use of Serpent (via ad units with increased Serpent taint) and growing conversion, though with significant variation across ad units. Keep in mind that other than Serpent, all of the ad unit's conversions will be due to legitimate user activity and not ZeroAccess. This result shows that the use of Serpent does add conversions of fraudulent click activity.

Next, to understand if the combined blending of Serpent and auto-clicking avoided the smart-pricing mitigations, we compare our sample of 54 dirty ad units to a random sample of ad units of similar size. We find that compared to a random sample, those

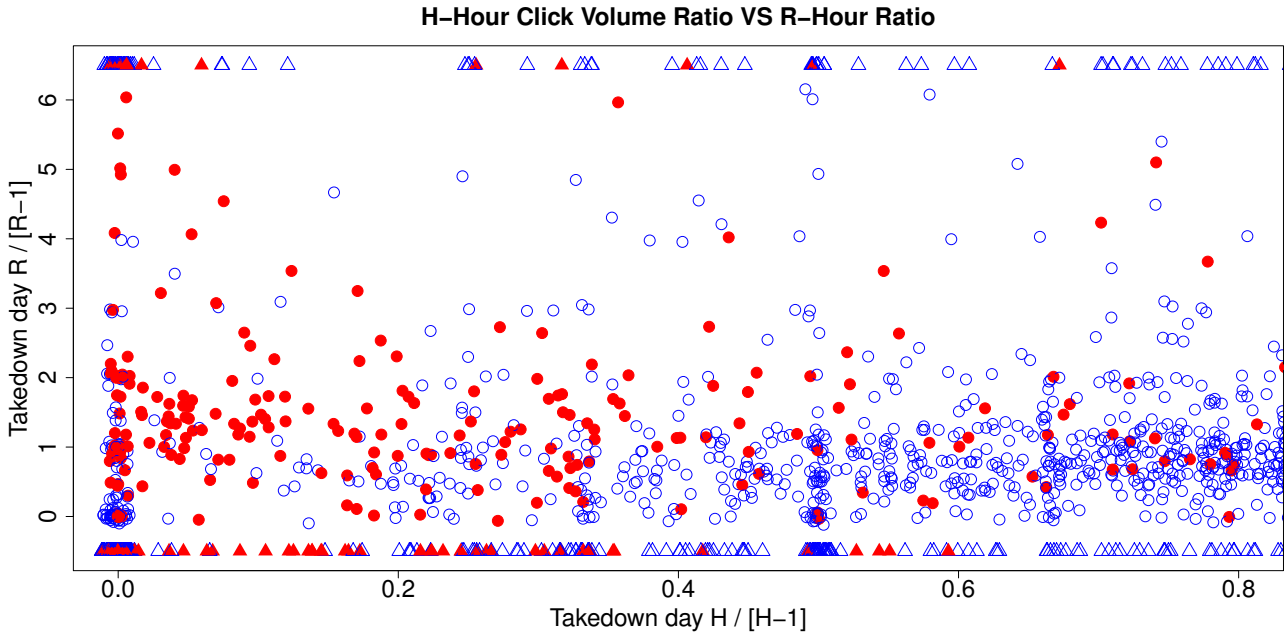

Figure 5: Ad unit response to Takedown (X axis, in terms of H-hour ratio) versus to subsequent dissemination of new auto-clicking module (Y axis, in terms of R-hour ratio). Solid red circles indicate ad units with ZA taint ≥ 0.4. Triangles below the Y origin reflect ad units that had no traffic during either $R-1$ or R hours. Triangles above $Y = 6$ had traffic during R-hour but not during $R-1$. Note that we have added small amounts of Gaussian jitter to prevent coincident points from completely overlapping.

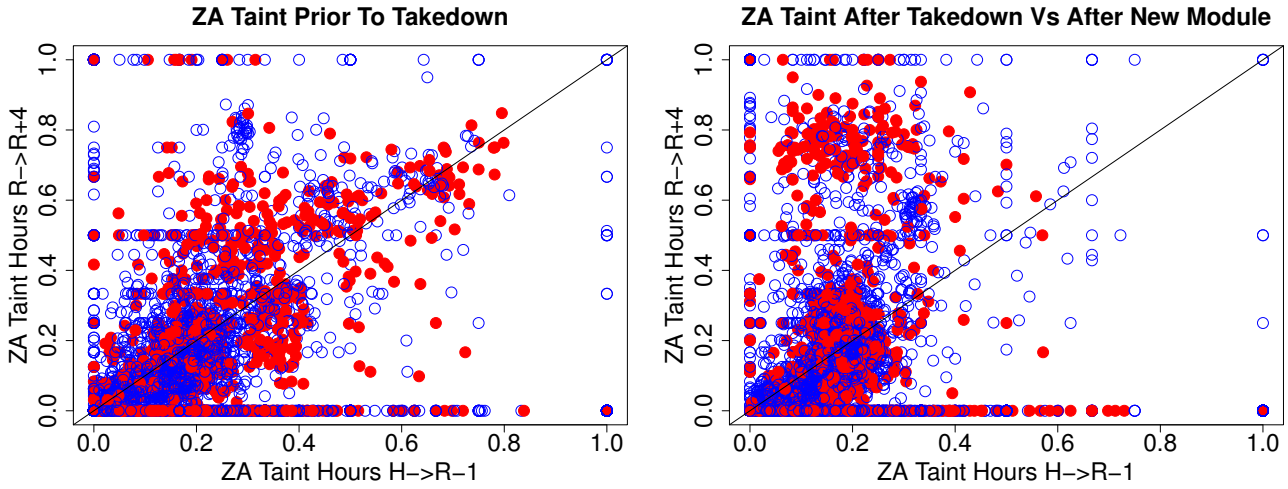

Figure 6: Comparison of ZA taint before and after R-hour, per ad unit. The X axis gives taint from between the Takedown hour and just before the hour corresponding advent of the new auto-clicking module post-Takedown; the Y axis as computed over the 4 hours starting with R-hour. The left plot shows the day prior to Takedown, the right plot the day of Takedown. Solid red circles reflect ad units that had a drop in H-hour ratio across the day before and after Takedown ≥ 0.5 (this corresponds to the population shift in Figure 3).

flagged as ZeroAccess dirty had conversions rates of up to three times less. As a result, these dirty ad units were indeed subject to the smart-pricing discount. Thus, it appears that if the intent of the Serpent module leveraging real users was to avoid smart-pricing, then it failed to do so since the users forced to go to the advertiser pages did not convert enough. If the intent behind leveraging real users, however, was simply to avoid detection while accepting the smart-pricing discounted income, then it was successful.

5.2 Estimates and Challenges

Formulating a sound estimate of the global impact of ZeroAccess proves very difficult, not only due to limitations in our data sets, but more fundamentally because of the sharply limited visibility that the ad network ecosystem provides. Using some plausible assumptions, we can derive estimates of ZeroAccess's impact perhaps within an order of magnitude, and note that these paint roughly a similar picture to that developed in the work of others [40]. Clearly, however, the important question of just which parties make how much profit due to this illicit activity poses thorny methodological challenges.

Limitations within our ad network. Although our ad network partner provided access to the complete click payout information for the 54 dirty ad units we identified, those ad units make up only a subset of the likely total ZeroAccess traffic abusing their network.

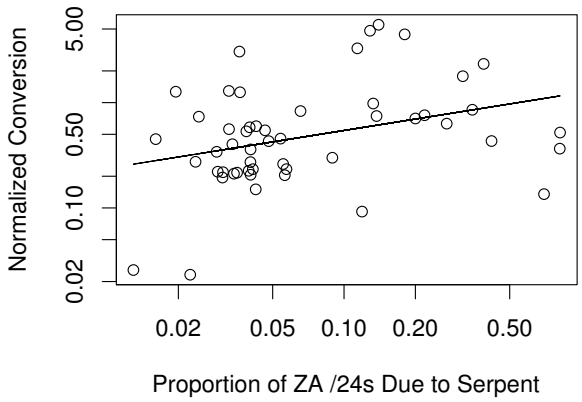

Figure 7: Relationship between the proportion of ZeroAccess traffic that includes Serpent and the conversion rate for the dirty ad units. The X axis gives the proportion of an ad unit's ZeroAccess subnets that also appeared in the DNS telemetry data, and thus include Serpent activity. The Y axis reflects the normalized conversion rate across the set (1.0 = the mean rate of the set). We log-scale both axes, discarding 3 ad units that had either no Serpent subnets or never converted. The line shows a least-squares fit to the log-transformed data, corresponding to a power-law relationship with an exponent of about 0.36.

Methodologically, we require an ad unit to exhibit a baseline level of traffic before we can label it as fraudulent (via manual analysis). Ad units, however, exhibit a very long tail in their level of activity. The vast majority provided (individually) so little traffic that although they exhibited numerous ZeroAccess characteristics, such as high ZeroAccess taint, for any given such ad unit we could not rule out these characteristics occurring simply by chance, and thus could not definitively label them as bad. In aggregate, however, such ad units potentially account for a large volume of traffic, much more than those of the 54 definitively dirty ad units that we identified. How to soundly attribute a portion of that traffic as fraudulent remains an open question.

Limitations across ad networks. A further complication arises in identifying the breakdown of fraud between our ad network partner versus other networks targeted by ZeroAccess. Our "milking" data provides a qualitative impression of what this balance might look like, finding that 367 of the 1,766 click chains transited our ad network partner. Superficially this suggests that in total, ZeroAccess click activity might be about five times what our ad network partner sees. However, along with the limited scale of the milking data, other basic factors, such as how presented ads (and thus click chains) are influenced by IP address or search term, could result in major shifts in this ratio. In addition, the milking data only reflects one instance of Serpent activity, and we do not know if its distribution of ad networks accords with that of the auto-clicking module.

Limitations in DNS data. While our DNS telemetry gives us visibility into Serpent ad clicks, the data suffers from significant noise. Moreover, we do not know how to extrapolate from pure Serpent activity to the behavior of the auto-clicking module. When merging the DNS data with other sources, we also encounter issues with both the granularity of the IP address information and IP address churn, making attribution within our partner's ad network problematic.

Assumptions and estimates. Given those major caveats, we now formulate some rough estimates to get a sense of the overall scale of activity.

The 54 dirty ad units we identified generated over 100K clicks per day. It appears certain that aggregate ZeroAccess activity far exceeds this lower bound, which we view as very solid (we have high confidence those ad units all reflect ZeroAccess activity) but also very conservative. Building on this foundation, given the large volume of suspicious-but-not-definitive ad units we investigated—both the "long tail" with dubious but sparse activity, and those that appear to carry ZeroAccess traffic blended with non-ZeroAccess traffic—we would argue that likely the total activity within our ad network partner was at least a factor of $2x$ larger.

If we then take the $5x$ factor seen in our limited click chain milking to represent the volume of traffic seen in other ad networks, we have a total extrapolation of $10x$ to our original conservative estimate, yielding 1M fraudulent clicks per day. This range is consistent with the order of magnitude of fraudulent clicks estimated from our DNS data.

According to our ad partner, the cost-per-click of this traffic after smart-pricing normalization would be between 10–30 cents. Taking the low end of these ranges, we can construct an estimate of the total click fraud impact of ZeroAccess as on the order of $100,000 per day, which aligns with the estimate from Sophos of ZeroAccess generating up to $2.7 million in click fraud per month [40].

We highlight, however, that enough uncertainties exist that our figure can only reflect a "best guess" estimate. We also note that given the available data sources, we lack the ability to gauge what fraction of this money makes it into the hands of the botnet owners after the long chain of (sub)syndicates take their cut.

5.3 Discussion

Subsyndication is a problem. Subsyndication is the key business model enabling large-scale click fraud. Even a single (trusted) publisher that is allowed to subsyndicate opens the flood gates for click fraud traffic (blended with other legitimate traffic) entering the ad network unwittingly through that publisher. Our Serpent milker encountered syndication chains as long as 13 domains deep. It is no surprise therefore that we were able to definitively conclude significant ZA dirtiness for only 54 ad units. There were many more where our analysis was inconclusive due to high levels of blending that diluted the otherwise sharp signal presented by the Takedown.

One might argue that *anonymous* subsyndication is the real problem, and if publishers could be forced to identify themselves, the search engine would be able to better police the ad network. While this is true, we feel it is impractical. The level of blending we see suggests that many publishers are complicit in click-fraud, especially since everyone along the syndication chain benefits financially from it. Even if the search engine could recursively force (sub)syndicates to set a policy that requires identifying their traffic sources—a hard problem in itself—there would be little reason for the (sub)syndicates to enforce it.

Low signal to noise ratio. Our analysis would not have been possible absent the strong signal injected into the ZeroAccess botnet by the Takedown. Even then we had to combine three large datasets from three different vantage points. Such strong signals are too few and far between to be an effective means of policing ad networks. The capability to inject more frequent (but perhaps less intense) disruptions into botnets could create a strong temporal signal that could serve as a basis for ongoing click fraud policing.

6. CONCLUSION

We have undertaken a detailed examination of the activity of one of the largest click fraud botnets in operation, ZeroAccess. Through reverse-engineering and controlled execution, we mapped out ZeroAccess's click fraud component, including the innovations it introduced in hijacking high-quality user search traffic. In the process we discovered side channels in the form of vestigial DNS

lookups that enabled us to track its large-scale activity. We then combined this perspective with partial "supernode" data capturing the botnet's peer-to-peer C&C activity in order to match up botnet activity data with ad-unit click stream data provided by our major ad network partner.

By leveraging the striking shifts induced in click activity by Microsoft's attempted takedown of ZeroAccess in Dec 2013, along with the ZeroAccess botmaster's subsequent response, we combined these diverse data sources to identify with high confidence 54 individual ad units whose traffic volume (and hence revenue) primarily derived from ZeroAccess. Our ensuing analysis of these ad units revealed that while the latest generation of click fraud botnets have circumvented many detection approaches, they are still mitigated (for now) by the smart-pricing mechanism. Extrapolating from the known-bad ad units, we constructed an estimate that ZeroAccess likely generated on the order of a million fraudulent clicks per day across all ad networks, with the overall ecosystem revenue diverted by the botnet's activity roughly on the order of $100,000 per day.

Finally, we note that the complexity of the online advertising ecosystem is such that no one party comes remotely close to having comprehensive visibility into who gets paid what for any given click. The hodgepodge of data sources required for our analysis starkly illustrates both the tangled nature of the click fraud problem space and the pressing need for much better mechanisms for correlating traffic and payment streams.

7. ACKNOWLEDGEMENTS

The authors are grateful for the assistance, contributions, and guidance of many individuals and organizations, including David Anselmi, Richard Boscovich, Hitesh Dharamdasani, Yunhong Gu Niels Provos, Moheeb Abu Rajab, Nicholas Weaver, the Google Safe Browsing Team, the Microsoft Digital Crimes Unit (DCU), the Microsoft Malware Protection Center (MMPC), and the Bing Ads Online Forensics Team.

Without such contributions, this work would not be possible.

This work was supported in part by National Science Foundation grants NSF-1237076, NSF-1237264, NSF-1237265, and CNS-0831535, by the Office of Naval Research MURI grant N00014-09-1-1081, and by generous support from Google, Microsoft, and the UCSD Center for Networked Systems (CNS).

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

8. REFERENCES

[1] G. Bonfa. Step-by-Step Reverse Engineering Malware: ZeroAccess / Max++ / Smiscer Crimeware Rootkit. http://resources.infosecinstitute.com/step-by-step-tutorial-on-reverse-engineering-malware-the-zeroaccessmaxsmiscer-crimeware-rootkit, November 2010.

[2] S. Contavalli, W. van der Gaast, Leach, and E. Lewis. Client Subnet in DNS Requests. https://datatracker.ietf.org/doc/draft-vandergaast-edns-client-subnet/, 7 2013. RFC Draft.

[3] N. Daswani and M. Stoppelman. The Anatomy of Clickbot.A. In *Proc. HotBots*, 2007.

[4] V. Dave, S. Guha, and Y. Zhang. Measuring and Fingerprinting Click-spam in Ad Networks. In *Proceedings of ACM SIGCOMM*, 2012.

[5] V. Dave, S. Guha, and Y. Zhang. ViceROI: Catching Click-Spam in Search Ad Networks. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2013.

[6] J. Dupre. What is Cloaking and Why Do Affiliate Marketers Use It? http://justindupre.com/what-is-cloaking-and-why-do-affiliate-marketers-use-it/, May 2010.

[7] B. G. Edelman. Google Click Fraud Inflates Conversion Rates and Tricks Advertisers into Overpaying. http://www.benedelman.org/news/011210-1.html, Jan. 2010.

[8] Federal Bureau of Investigation. International Cyber Ring That Infected Millions of Computers Dismantled. http://www.fbi.gov/news/stories/2011/november/malware_110911, Nov. 2011.

[9] M. Giuliani. ZeroAccess, an advanced kernel mode rootkit. http://www.prevx.com/blog/171/ZeroAccess-an-advanced-kernel-mode-rootkit.html, Apr. 2011.

[10] Google Ads: Ad Traffic Quality Resource Center Overview. http://www.google.com/ads/adtrafficquality/.

[11] Google Inc. About smart pricing. *AdWords Help*, Apr. 2013.

[12] Google Inc. Google Services Agreement for InfoSpace LLC. http://www.sec.gov/Archives/edgar/data/1068875/000119312514121780/d702452dex101.htm, March 2014.

[13] Google Inc. How Google uses conversion data. *AdWords Help*, May 2014. https://support.google.com/adwords/answer/93148.

[14] T. Greene. ZeroAccess bot-herders abandon click-fraud network. http://www.networkworld.com/news/2013/121913-zeroaccess-277113.html, Dec. 2013.

[15] C. Grier et al. Manufacturing Compromise: The Emergence of Exploit-as-a-Service. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, October 2012.

[16] H. Haddadi. Fighting Online Click-Fraud Using Bluff Ads. *ACM SIGCOMM Computer Communication Review*, 40(2):22–25, Apr. 2010.

[17] F. Howard. Exploring the Blackhole exploit kit. http://nakedsecurity.sophos.com/exploring-the-blackhole-exploit-kit/.

[18] HSI seizes Silk Road underground black market website. http://www.ice.gov/news/releases/1310/131002baltimore.htm, 2013.

[19] P. Ipeirotis. 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, Mar. 2011.

[20] Kaffeine. MagicTraffic : a look inside a Zaccess/Sirefef affiliate. http://malware.dontneedcoffee.com/2013/11/magictraffic-look-inside-zaccesssirefef.html, 2013.

[21] L. Kim. The Most Expensive Keywords in Google AdWords. http://www.wordstream.com/blog/ws/2011/07/18/most-expensive-google-adwords-keywords, July 2011.

[22] 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. In *International Symposium on Software Reliability Engineering*, 2009.

[23] B. Krebs. Fake Antivirus Industry Down, But Not Out. http://krebsonsecurity.com/2011/08/fake-antivirus-industry-down-but-not-out/, August 2011.

[24] B. Krebs. Reports: Liberty Reserve Founder Arrested, Site Shuttered. http://krebsonsecurity.com/2013/05/reports-liberty-reserve-founder-arrested-site-shuttered/, 2013.

[25] B. Krebs. ZeroAccess Botnet Down, But Not Out. http://krebsonsecurity.com/tag/zeroaccess-takedown/, Dec. 2013.

[26] The Lote Clicking Agent. http://www.clickingagent.com/.

[27] D. McCoy, H. Dharmdasani, C. Kreibich, G. M. Voelker, and S. Savage. Priceless: The role of payments in abuse-advertised goods. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. ACM, 2012.

[28] K. McNamee. Malware Analysis Report. Botnet: ZeroAccess/Sirefef. http://www.kindsight.net/sites/default/files/Kindsight_Malware_Analysis-ZeroAcess-Botnet-final.pdf, February 2012.

[29] A. Metwally, D. Agrawal, and A. El Abbadi. DETECTIVES: DETEcting Coalition hiT Inflation attacks in adVertising nEtworks Streams. In *WWW Conference*, 2007.

[30] A. Metwally, F. Emekçi, D. Agrawal, and A. El Abbadi. SLEUTH: Single-pubLisher attack dEtection Using correlaTion Hunting. In *VLDB*, 2008.

[31] Microsoft, Yahoo! Change Search Landscape. http://www.microsoft.com/en-us/news/press/2009/jul09/07-29release.aspx, July 2009.

[32] 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, 2011.

[33] A. Neville and R. Gibb. ZeroAccess Indepth (Symantec Corporation White Paper). http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/zeroaccess_indepth.pdf, October 2013.

[34] P. Pearce, C. Grier, V. Paxson, V. Dave, D. McCoy, G. M. Voelker, and S. Savage. The ZeroAccess Auto-Clicking and Search-Hijacking Click Fraud Modules. Technical report, EECS Department, University of California, Berkeley, Dec 2013.

[35] Pricewaterhouse Coopers. IAB Internet Advertising Revenue Report: 2013 First Six Months' Results. http://www.iab.net/media/file/IAB_Internet_Advertising_Revenue_Report_HY_2013.pdf, October 2013.

[36] E. Rodionov and A. Matrosov. The Evolution of TDL: Conquering x64. http://go.eset.com/us/resources/white-papers/The_Evolution_of_TDL.pdf, 2011.

[37] C. Rossow, D. Andriesse, T. Werner, B. Stone-Gross, D. Plohmann, C. J. Dietrich, and H. Bos. SoK: P2PWNED — Modeling and Evaluating the Resilience of Peer-to-Peer Botnets. In *IEEE Symposium on Security and Privacy*, May 2013.

[38] L. Sinclair. 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, May 2011.

[39] A. Tuzhilin. The Lane's Gifts v. Google Report. http://googleblog.blogspot.com/pdf/Tuzhilin_Report.pdf, 2005.

[40] J. Wyke. The ZeroAccess Botnet: Mining and Fraud for Massive Financial Gain. http://www.sophos.com/en-us/why-sophos/our-people/technical-papers/zeroaccess-botnet.aspx, September 2012.

[41] J. Wyke. ZeroAccess. http://www.sophos.com/en-us/why-sophos/our-people/technical-papers/zeroaccess.aspx, 2012.

[42] F. Yu, Y. Xie, and Q. Ke. SBotMiner: Large Scale Search Bot Detection. In *WSDM*, 2010.