Dissecting Ghost Clicks: Ad Fraud Via Misdirected Human Clicks

Sumayah A. Alrwais*
Indiana University
Bloomington, U.S.A.
salrwais@cs.indiana.edu

Christopher W. Dunn
Indiana University
Bloomington, U.S.A.
chrdunn@cs.indiana.edu

Minaxi Gupta
Indiana University
Bloomington, U.S.A.
minaxi@cs.indiana.edu

Alexandre Gerber
AT&T Labs-Research, U.S.A.
alexgerber67@gmail.com

Oliver Spatscheck
AT&T Labs-Research, U.S.A.
spatsch@research.att.com

Eric Osterweil
Verisign Labs, U.S.A.
eosterweil@verisign.com

ABSTRACT

FBI's *Operation Ghost Click*, the largest cybercriminal takedown in history, recently took down an ad fraud infrastructure that affected 4 million users and made its owners 14 million USD over a period of four years. The attackers hijacked clicks and ad impressions on victim machines infected by a DNS changer malware to earn ad revenue fraudulently. We experimented with the attack infrastructure when it was in operation and present a detailed account of the attackers' modus operandi. We also study the impact of this attack on real-world users and find that 37 subscriber lines were impacted in our data set. Also, 20 ad networks and 257 legitimate Web content publishers lost ad revenue while the attackers earned revenue convincing a dozen other ad networks that their ads were served on websites with real visitors. Our work expands the understanding of modalities of ad fraud and could help guide appropriate defense strategies.

1 Introduction

Online advertising is a fast growing multi-billion dollar industry. At its core are content *publishers*, such as websites that provide news and search functionality, and *advertisers* which are connected to each other by *advertising networks*. The advertisers pay the advertising networks for displaying their advertisements (or ads, as they are referred to in common parlance). The ad networks act as brokers, finding suitable publishers for ads and then splitting the revenue with them. Common revenue models include: cost per mille (CPM), where advertisers are charged per thousand impressions; cost per click (CPC), where advertisers are charged per click; and cost per action (CPA), where advertisers are charged per action, such as an online sale.

As revenues generated by online advertising have increased, so has the fraudulent activity. Examples of fraudulent activity include publishers generating unwarranted ad revenue through the use of human clickers or botnets. Publishers are also known to attract higher value ads by manipulating keywords on their websites. Further, ill-intended publishers sometimes stack ad impressions on top of each other to collect revenue hurting the advertisers of these ads. On the other hand, advertisers are also known to serve malicious ads and to also hurt their rivals by exhausting their advertising budget [15, 11].

Owing to the seriousness of the problem, a few recent works have examined how ad fraud is perpetrated. Miller et al. investigated a botnet that was used to commit ad fraud and then used that understanding to study how ad networks can identify and combat ad fraud [35]. Complementary work by Stone-Gross et al. studied the operation of two malware families, *Fiesta* and *7cy*, whose bots are known to commit ad fraud by clicking on ads [31]. A common theme in these works are ensembles of *clickbots*, botnets that specialize in committing ad fraud. Zhang et al. recently conducted a study that verified the role of human clickers in ad fraud [42]. Our work expands the landscape of ad fraud beyond these works and proves that ad fraud needs to be understood better in order for the defenses to be adequate.

Specifically, in this paper, we present the dissection of an ad fraud scheme whose infrastructure was recently taken down by the FBI under *Operation Ghost Click*. It was the largest cybercriminal takedown in history [27, 34, 36] that made its attackers 14 million USD over a course of four years. The attackers did not use any clickbots. Instead, real human clicks were misdirected to generate ad revenue fraudulently. The key element in this scheme was a DNS changer malware that changed the DNS resolver setting on victim machines to attacker-controlled resolvers in Eastern Europe. This simple change allowed the attackers to hijack ad impressions and clicks and earn CPM and CPC revenue, all the while fooling tens of ad networks that the revenue was earned on publisher websites owned by the attackers. A subtle distinction of this ad fraud scheme with respect to clickbots was that while clickbots only hurt advertisers, this scheme earned the attackers revenue at the cost of hurting many legitimate publishers, ad networks and advertisers. The key contributions of our work are the following:

- In situ experimentation: We experimented with the attack infrastructure when it was alive and in-action. This allowed us to scrutinize the attackers' modus operandi in depth. We present a detailed account of the operation.

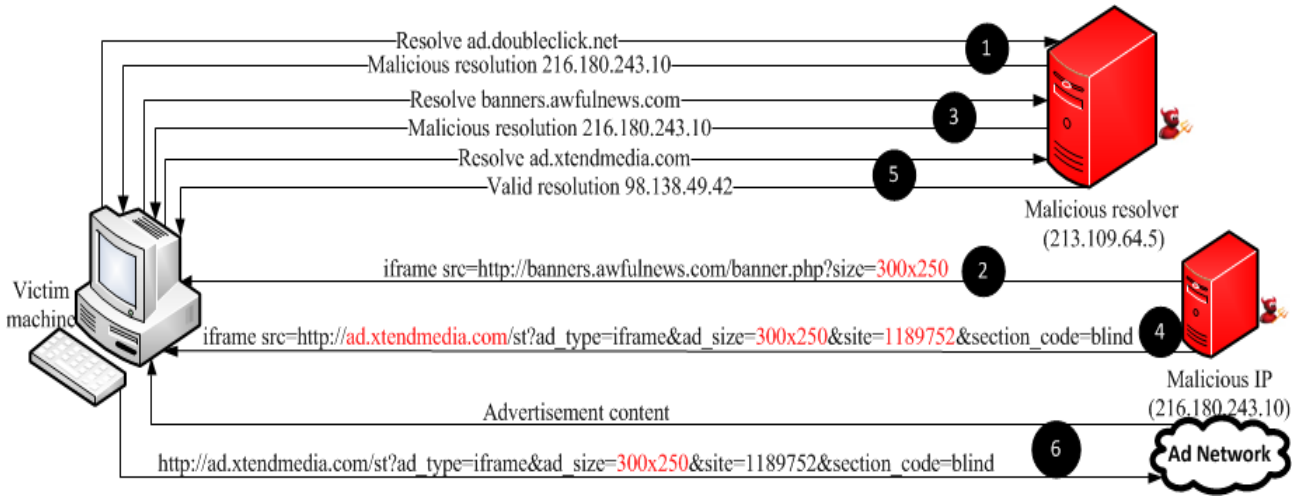
- Mapping the attack infrastructure: We develop a methodology to map the attack infrastructure and find that a total of 1039 malicious DNS resolvers were misdirecting

*Funded by the College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia.
salrwais@ksu.edu.sa

ACSAC '12 Dec. 3-7, 2012, Orlando, Florida USA

```
http://ad.doubleclick.net/adi/ebay.ebayus.homepage.cchp/cchp;sz=300x250;ord=1305150844507;u=i_
9083996730633540328|m_172562;dcopt=ist;tile=1;um=0;us=13;eb_trk=172562;pr=20;xp=20;np=20;uz=;
```

(a) An ad URL on eBay website



(b) Attack chain leading to ad replacement

Figure 1: Steps in the ad replacement attack vector

victims of the DNS-changer malware and the attackers used 30 different IP addresses to commit fraud. These IP addresses were registered with 6 different ISPs. Overall, the attack infrastructure was well provisioned, with attention paid to fault tolerance.

- Gauging attack impact: Using HTTP traces of 17K DSL subscribers of a Tier-1 ISP for a day, we estimate the impact of this fraud. We find that 20 ad networks and 257 legitimate content publishers lost revenue. The attackers stole this revenue by pretending to be a publisher of 423 websites with at least a dozen different ad networks they defrauded. Our estimates of affected users come close to the numbers quoted by the FBI.

-Mitigation: Our work highlights that scrutinizing open recursive resolvers, a characteristic shared by the malicious resolvers used in the attack, could be a fruitful dimension in the fight against ad fraud.

2 Ad Fraud Scheme

The attackers made use of two attack vectors to earn ad revenue fraudulently. In the first, they earned cost per mille (CPM) revenue by replacing displayed ads. In the second, they earned cost per click (CPC) revenue by hijacking clicks. Both the attack vectors worked on victim machines infected by a DNS changer malware [37, 7, 19], which is a malware that utilizes social engineering to download and install a certain codec to play a video. Upon installation, it changes the machine's local DNS resolver to a malicious one. Different variants of DNS changers have surfaced, such as ZCodec [40]. Another way of altering the DNS settings is by using a rogue DHCP which sends the IP address of a rogue DNS server along with the allocated IP to any machine that requests to connect to the network [38, 21]. Yet another variant of the DNS changer attacks routers to change their resolver settings [30]. The latest version of this malware that appeared in 2011 is the Google redirect malware [32] which was reported to hijack Google's search engine results.

To carry out these attacks, the attackers' registered many websites they owned as *publisher* websites with various ad networks. We refer to the attacker-controlled websites as *front-end websites* subsequently in this paper. They also controlled a network of malicious DNS resolvers in Eastern Europe that victim machines contacted instead of legitimate resolvers, say ones provided by their ISPs. The malicious DNS resolvers selectively mis-resolved hosts that serve ads or Javascripts, enabling the two attack vectors. Next, we explain each attack vector through a real-world example.

2.1 Ad Replacement Attack

Consider a victim machine visiting a web page, `ebay.com`. The visit generates an HTTP request for an ad from DoubleClick using the URL shown in Figure 1(a). The ad host in this URL is `ad.doubleclick.net`. Figure 1(b) shows the steps that ensue:

Step-1: The malicious resolver mis-resolves the ad host name, `ad.doubleclick.net`, to `216.180.243.10`.

Step-2: The malicious IP serves an iFrame to the victim. The iFrame directs the victim to a front-end website to serve an ad of the same size as the original ad.

Step-3: The front-end website, `banners.awefulnews.com`, is also mis-resolved to the malicious IP address `216.180.243.10`. The mis-resolution ensures that anyone scrutinizing the front-end website, say an ad network, can be shown legitimate-looking content through a correct resolution.

Step-4: The mis-resolved front-end website serves an iFrame. The iFrame loads an ad URL which belongs to a different ad network, `XtendMedia`. The attackers have an account with this ad network (value of the "site" parameter in the Figure) and have convinced the ad network that it will display its ads on the front-end website.

Step-5: The new ad network's host name, `ad.xtendmedia.com`, is resolved correctly and it serves an ad.

Step-6: The new ad network serves an ad to the victim which the malicious resolver resolves correctly. The publisher ID on the displayed ad URL is one of the attackers' IDs with that ad network.

At the end of the above series of steps, the attackers collected CPM revenue for displaying an ad from `XtendMedia`

#	Method	Status	URL	IP	Notes
1	GET	200	http://www.google.com/s?YYYY&**q=download antivirus free**& YYYY	74.125.159.106	Legitimate host names correctly resolved
2	GET	200	http://www.google.com/url?sa=t&YYYY&**url=http://free.avg.com**& YYYY&**q=download antivirus free**& YYYY	74.125.159.107	
3	GET	200	http://free.avg.com/us-en/homepage	63.236.253.27	
4	GET	200	http://www.google-analytics.com/ga.js	205.234.201.229	Legitimate host name mis-resolved
5	GET	200	http://search2.google.com/123.php?ref=http://www.google.com/url?sa=t& YYYY&**url=http://free.avg.com**&YYYY&**q=download antivirus free**&YYYY	67.210.14.53	Non-existent host names
6	GET	200	http://search3.google.com/?**_kwd=downloadantivirusfree**& **lnk=http://free.avg.com/us-en/homepage**&**t=g**		
7	GET	200	http://bulletindaily.com/?65287254d575354084f584e49		Front-end website mis-resolved
8	GET	200	http://65.60.9.238/click.php?c=eb951aa106822a13341f45005500	65.60.9.238	Form click IP
9	POST	302	http://www.accurately-locate.com/jump2/? affiliate=6990&subid=191&**terms=download antivirus free**	67.29.139.153	Fake search engine correctly resolved
10	GET	302	http://r.looksmart.com/og/YYYYad=753242267;YYYY qt=downloadantivirusfreeYYYYsubid=6990.191;YYYY rh=accurately-locate.com|http://buy.norton.com/YYYY|referrer	74.120.237.11	Search ad network correctly resolved
11	GET	302	http://buy.norton.com/YYYY**68998-6990.191**| www.accurately-locate.com/jump1?affiliate=6990& subid=191&terms=downloadantivirusfree& YYYY	166.98.228.51	Legitimate host name correctly resolved

Table 1: Example of organic search click hijacking: The victim intended to visit `free.avg.com` and is instead taken to a sponsored result from search ad network, `looksmart.com`, for `buy.norton.com`. The occurrence of pattern "YYYY" in the URL indicates a truncation for brevity sake.

on one of their front-end websites. The original ad network, `DoubleClick`, and publisher website victim visited, `eBay.com`, lose ad revenue that they were expecting to earn. This attack is stealthy from victim's perspective since the content on the website they visit is unchanged. Further details of the attack as described in Section 4.

2.2 Click Hijacking Attack

There are four different modes for this attack vector. We describe one here. The rest are outlined in Section 5. In this mode, the victim's click on an organic search engine's result is hijacked and the victim is redirected to another website. Table 1 shows the steps that ensue:

Step-1: The victim visits search engine, `google.com`, through a malicious resolver, which resolves it correctly. Victim's search terms, "download free anti virus", return results.

Step-2: The victim clicks on an organic search result, `free.avg.com`, which is also resolved correctly.

Step-3: The victim visits `free.avg.com`, which is resolved correctly.

Step-4: The site, `free.avg.com`, attempts to load a Google Analytics script, `ga.js`. The malicious resolver mis-resolves the host name for this script, `www.google-analytics.com`, to inject a script into the parent document.

Step-5: The injected script, `123.php`, is loaded from a non-existent host name, `search2.google.com`, which is resolved by the malicious resolver to another malicious IP address. The injected script gets the referrer, `www.google.com`, search terms entered by the victim, as well as `free.avg.com` from `ga.js` and sends it to another non-existent host name, `search3.google.com`.

Step-6: The host, `search3.google.com`, hijacks the click if the referrer indicates that the victim came from a search engine. Otherwise, an empty response is returned. Since the referrer in this case was `google.com`, the click is hijacked by returning an HTML frame that sends the victim to an attacker-controlled front-end website, `bulletindaily.com`.

Step-7: When the victim visits the front-end website, the malicious resolver mis-resolves it to guard against external scrutiny just as in the case of ad replacement attack. The front-end website receives the search terms encoded. It contains a form to submit them to another malicious IP upon changing the encoding.

Step-8: The malicious IP, `65.60.9.238`, chooses to monetize the click by passing it through an affiliated fake search engine using an HTTP redirect.

Step-9: The fake search engine, `accurately-locate.com`, has access to the victim's search terms. It uses them, along with the attackers' affiliate ID, to generate a click on one of its sponsored results. The ad is retrieved from a search ad network, `looksmart.com`.

Steps-10-11: The ad network, `lookSmart.com`, redirects the victim to `buy.norton.com`. The referrer field for both steps is `bulletindaily.com`.

In this example, the attackers monetized a user's click on an organic search engine's result by selling the click through a fake affiliate search engine. The page, `free.avg.com`, was loaded temporarily before being redirected to `buy.norton.com`. The attackers defrauded `avg.com` by stealing its traffic. Search ad network, `LookSmart`, and `Norton` were defrauded by simulating a click on `Norton's` textual ad and collecting CPC revenue from `LookSmart` through its affiliate fake search engine. We observed clicks being monetized through one of three kinds of intermediaries: ad networks, ad exchanges and affiliate marketing programs.

3 Identifying When Resolvers Lie

We started our investigation with two IP addresses of malicious resolvers in the 213.109.0.0/20 prefix which were given to us by a Trend Micro researcher involved in helping the FBI with Operation Ghost Click. We probed these malicious resolvers to infer when they lied about ad host names. Toward that goal, we visited top 3,000 websites according to Alexa [1] on May 11, 2011 and extracted ad URLs by running the captured HTTP traffic against the ad URL pattern matching list [6] used by Adblock Plus, a popular ad blocking tool. Since the starting point of most ads today are Javascripts or HTML URLs which subsequently include or generate URLs to other ad components, such as images, we focused only on Javascripts and HTML URLs to identify ad hosts. The resulting data set contained a total of 7,483 unique HTML and Javascript ad URLs which were delivered by 1,019 ad hosts. We queried the malicious resolvers for these host names and applied the following two filtering heuristics to determine when a host name was mis-resolved.

Note also that our filtering heuristics are intentionally generous, in that they look for any explanation that may justify a resolution returned by a malicious resolver as correct before labeling it as incorrect. Consequently, we might underestimate which ad host names malicious resolvers lie about but are unlikely to penalize a good resolution from a malicious resolver as incorrect.

- Heuristic 1: Resolution contains a valid IP address: For each ad host name resolved by a malicious resolver, this filter checks if any of the IP addresses were also returned by a good DNS resolver. Since DNS resolutions may be influenced by the geographical location of the resolver, such as when a content distribution network (CDN) is in use, or load balancing considerations, we gathered good DNS resolutions from 4,490 public resolvers around the world covering 74 countries [8]. Each public resolver in the list was queried for the `A` record of all ad host names multiple times until it stopped responding with new IP addresses. Specifically, if an IP address returned by a malicious resolver was returned by a public DNS resolver for any ad host name, this heuristic considers all IP addresses in that resolution to be good.

Running this heuristic on 2,952 IP addresses returned by the malicious DNS resolvers for the 1,019 ad hosts cut down the set of suspicious IP addresses by 90.5% in that 281 IP addresses remained unverified by this filter. These IP addresses corresponded to 96 host names.

- Heuristic 2: Suspicious IP returns a valid SSL certificate: This heuristic leverages that many ad networks support secure Web-based logins for their advertisers for tasks such as updates to their ad campaign and payment. To apply this heuristic, we established an HTTPS connection with each IP address of a valid resolution for the 96 host names remaining to be verified from Heuristic 1 in order to determine if the corresponding host names provided secure logins. We found that over 98% of the valid IPs corresponding to 62 host names returned a valid certificate. (The remaining 2% are attributable to connection failures or maintenance issues.) Upon examining the suspicious resolutions of the 62 host names, we found **8** suspicious IP addresses that did not return a certificate. We took this to indicate that these 8 IP addresses were malicious. These 8 malicious IPs corresponded to four host names. In fact, they were hosting an additional 23 host names, bringing the number of mis-resolved host names to 27. The list of mis-resolved host names included `ad.doubleclick.net`, `view.atdmt.com`, `tag.admeld.com` and `pagead2.googlesyndication.com`, each of which are popular ad hosts.

In analyzing the DNS records (`A`, `NS`, `SOA`) of these 8 malicious IPs, we observed common features. For example, we saw a typical TTL of 600 and `NS` & `SOA` records that were consistent.

4 Aspects of Ad Replacement

4.1 Ad Networks Defrauded

Section 3 helped identify 27 ad host names malicious resolvers were lying about. There were 1,277 Javascript and HTML ad URLs that belonged to them. In order to understand how the attackers were carrying out ad replacement, we setup a test machine to use a malicious resolver as its primary DNS resolver and visited each of the 1,277 ad URLs. The process helped identify how attackers were earning CPM revenue, as described in Section 2.1. We also learnt that the attackers had accounts with three ad networks shown in Table 2. We saw a total of four ad hosts from these networks and 34 different publisher IDs used by the attackers. The Table also shows the `CNAME` records for these host names, which we found by querying good DNS resolvers. The CNAMEs indicate that Redux Media and Xtend Media are both managed by Yahoo!.

Ad Network	Ad host names	CNAME	Publisher IDs
Redux Media	`ad.reduxmedia.com`	`ad.YM.com`	9
	`ads.reduxmediagroup.com`	`ib.adnxs.com`	23
Xtend Media	`ad.xtendmedia.com`	`ad.YM.com`	1
Clicksor	`ads.clicksor.com`		1

Table 2: Ad networks the attackers contracted with (YM stands for `yieldmanager`)

4.2 Operational Details

Out of the 1,277 ad URLs we tested, the attackers only successfully executed 782 URLs. In trying to answer why, we learnt a few interesting operational details about the attack:

- When unable to parse an ad URL, attackers loaded the original ad: We noticed that the size of the ad played an important role in determining whether an ad could be replaced or not. For example, in the ad URL shown in Figure 1(a), we can see that the size of the ad is `300x250` pixels which is clearly supported by `Xtend Media`, as evident from the completion of the attack chain. On the other hand, if the attackers encountered an ad size they could not support, they returned an empty response. As a concrete example, using the same ad URL as in Figure 1(a) with a modification of the size variable to `300x50` would cause the attackers to return an empty response. We found they supported 18 common size variations such as `160x600` and `728x90` while sizes `120x60` and `150x150` were not supported.

In cases where the attackers either did not support the ad type or could not extract size from the ad URL, the initial mis-resolution of the ad host redirected the victim to the intended ad server so that the original ad could be displayed. For example, if the path in the ad URL shown in Figure 1(a) was modified slightly by replacing `adi` with `ad`, as `http://ad.doubleclick.net/ad/ebay.ebayus.homepage.cchp` , the attackers would redirect the victim to `http://ad2.doubleclick.net/ad/ebay.ebayus.homepage.cchp`. Note that this modification is specific to ad host, `ad.doubleclick.net`. Our version of the edited URL specifies that the ad request is for an image (denoted by `/ad/`) not an iFrame (denoted by `/adi/`), as was the case with the original URL in Figure 1(a) which was supported by the attackers. Upon redirection to `ad2.doubleclick.net`, the attacker correctly resolves the host name to allow the originally intended ad to be loaded. The redirection helps the attackers escape detection from the ad network whose ad they were replacing. We observed the same redirection behavior for other ad hosts as well.

```
http://pagead2.googlesyndication.com/pagead/show_ads.
js?1306433500880
```

Figure 2: An example ad URL without size information. The identifier at the end is not size related.

- Attackers missed ad replacement opportunities: While examining URLs, we noticed cases where the attackers redirected the victim to the original correctly resolved ad host, causing them to miss out on additional profit. Specif-

ically, we found that ad sizes were parsed in a case-sensitive manner causing them to miss for example `300X250`, where the attackers were parsing only sizes with a small 'x'. Additionally, they were failing to recognize minor variations in URL paths and minor ad size variations where they were only replacing ads with exact sizes. For example, an ad URL with a size of `300x251` was not replaced with an ad of size `300x250`, where the later was supported by the attackers.

5 Modes of Click Hijacking

There are four different modes of the click hijacking attack, shown in Figure 3. We described one in Section 2.2. Here, we outline them all.

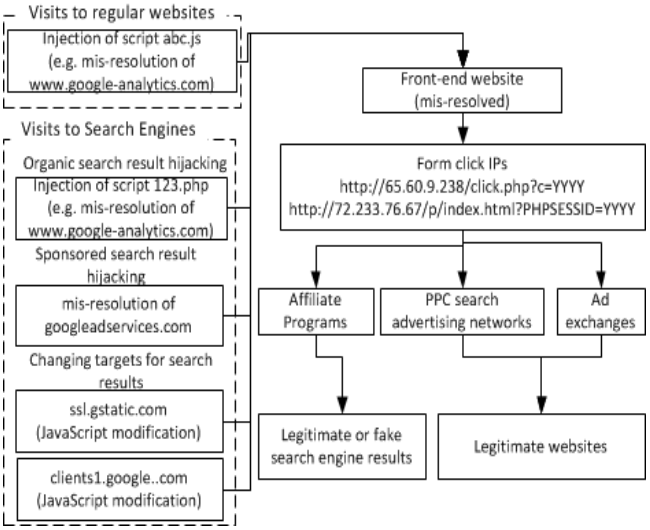


Figure 3: The four modes of the click hijacking attack

5.1 Visits to Search Engines

Most search engines display two types of results: *organic* and *sponsored*. While organic results appear because of their relevance to the search terms entered by a user, sponsored results are ads from advertisers who have contracted with the search engines. Search engines often attempt to display the most relevant sponsored results in the interest of luring visitors into clicking on them since clicks generate money for them.

In three of the modes of click hijacking, the attackers exploited the fact that most Internet users visit web pages through search engines and monetized their clicks. Broadly speaking, the attackers were either changing the targets of results displayed by a search engine or were leaving the targets as is but were hijacking clicks victims made on pages they visited through search engines. We outline each of the three modes below.

5.1.1 Organic search result hijacking

In the first mode, the attackers were monetizing clicks on organic search results, as detailed in Section 2.2. They were monetizing clicks through the injection of a server-side script, `123.php`, which was injected any time the victim clicked on an organic search result page containing a Javascript. We saw concrete instances of the injection on pages with the Google Analytics Javascript, `ga.js`, and when the attacker successfully replaced an ad involving a Javascript.

After hijacking a click, the victim was always directed to a front-end website which contains a form to submit search terms to one of a set malicious IP addresses. The form was submitted using the same search terms as the ones entered by the victim in the search engine. (Further testing revealed cases where the attacker was changing the keywords. In one instance, the keywords "download antivirus free" were changed to "best digital cameras".) We refer to this set of IP addresses as *form click* IPs in this paper. The form click IP addresses decided on the method of the click monetization by redirecting the user to one of the following options:

- Monetize through a pay-per-click (PPC) search advertising network: Many search engines and web portals with search functionality display text-based sponsored ads from PPC search advertising networks. A click on one of those ads redirects the user to the PPC search ad network, who does accounting based on the publisher's ID and search terms. We witnessed clicks being sold to PPC search ad networks such as `looksmart.com`, `admanage.com`, `adknowledge.com` and `dsidemarketing.com`.

- Monetize through an ad exchange: Ad exchanges are technology platforms that facilitate bidded buying and selling of online ads from multiple ad networks. We saw instances where a click on `free.avg.com` was sold through the `bidsystem.com` ad exchange to advertiser, `stopzilla.com`. We also witnessed other ad exchanges exploited in a similar way such as `qualibid.com`.

- Monetize via affiliation with search engines: In this form of monetization, the attackers set affiliate accounts with search engines and led victims to it. The latter displayed results for the keywords entered by the victim. The search engines paid the attackers based on their affiliate IDs. We found two types of search engines being used, legitimate and fake. The *fake search engines* were not directly controlled by the attackers. They displayed sponsored results through PPC search ad networks and clicks on these sponsored results were used either as-is or redirected to a malicious site depending on the affiliate who sent them the traffic. We also found instances using legitimate search engines where our clicks on `free.avg.com`, in the example described in Section 2.2, led to a set of search results displayed on a legitimate search engine, `star.feedsmixer.org`. We witnessed the attacker exploiting other search engines such as `infomash.org` and `info.com`, the same way.

5.1.2 Sponsored search result hijacking

In the second mode of the click hijacking attack, the attackers were hijacking clicks on sponsored search results in a manner similar to that for organic search results. Specifically in the case of Google, clicks on the sponsored search results were processed through `googleadservices.com`. The attackers were mis-resolving this host name to a new, dedicated malicious IP address which then ultimately led to a dedicated front-end website, `relited.com`. This website had a couple of press releases that appeared to be propagandas to establish its legitimacy in case ad networks investigated it for the clicks attackers were cashing. Once the victim reached the front-end website, it was redirected to one of the form click IPs to monetize this click in one of the three ways described in Section 5.1.1.

5.1.3 Changing targets for search results

In contrast to hijacking actual clicks on organic or sponsored search results, in the third mode of the attack, the attackers employed an alternative mode of hijacking human clicks. In this mode, they changed the targets for sponsored as well as

organic search results. We explored this attack in detail only in the context of the Google search engine but have preliminary evidence that the attackers were exploiting other popular search engines similarly. The attack essentially worked by mis-resolving the host names for two of the Javascripts used by Google in displaying search results. The first script is loaded from `clients1.google.com` and is used for autocompletion during typing of keywords. The second is loaded from `ssl.gstatic.com`, which is a Google host dedicated to static content, including images, Javascripts and CSS files. The attackers mis-resolved each of these two host names to two dedicated malicious IP addresses each and changed the functionality of these scripts by adding a snippet of code which resets click event handlers, such as `onclick` and `onmousedown` to cancel out the action originally set by Google and changes where the links in the returned HTML document lead. The modified links lead to non-existent host names with an argument list containing the original link returned by Google, keywords and referrer fields. Once the victim reached there, it was redirected to one of the form click IPs to monetize this click in one of the three ways described in Section 5.1.1.

5.2 Visits to Regular Websites

In the fourth and final mode of the click hijacking attack, the attackers were monetizing visits to regular websites that may not have originated at a search engine. This mode of attack was executed any time they managed to inject a Javascript, `abc.js`, using the same method of injection for the `123.php` script. Correspondingly, upon injection of the script, the attackers choose between loading `123.php` and `abc.js` randomly.

The script, `abc.js`, set two types of event handlers: 1) body onclick and 2) `<a>` link onclick. The script exploited only the first click of any of these events to avoid frustrating the victim to the extent that they would take immediate steps to get their machines cleaned up. As it is the mean time to clean the DNS-changer malware was 6-12 days [20]. The goal of this script was to open a new window in response to the click event. The new window belonged to non-existent host names from reputable domains and the attackers generated these host names based on what type of Javascript led to the injection of the malicious script. For example, if `abc.js` was injected through the Google Analytics Javascript, `ga.js`, then the new window was loaded from `search.google-analytics.com`. Since these host names were resolved by the attacker-controlled malicious resolvers, the attackers could choose any. Carrying the `search.google-analytics.com` example further, a visit to the URL in the new window ultimately led to one the dedicated front-end websites controlled by the attackers. The front-end websites were mis-resolved as before. The front-end websites dedicated to the `abc.js` script resolved to a group of three malicious IP addresses. Ultimately, the click was monetized in one of the three ways described in Section 5.1.1.

6 Attack Infrastructure

The attack infrastructure had three components. The first were the malicious resolvers. The second were the malicious websites the attackers used to earn CPM and CPC revenue. The third were the malicious IP addresses used in executing the attack chains. While our experiments revealed some of each, we suspected that there were more. In this section, we attempt to find the rest.

6.1 Malicious Resolvers

We started this study with the IP addresses of two malicious resolvers. Upon searching for online articles reporting on the behavior of malicious DNS resolvers [18, 32, 26], we found several IP addresses belonging to six IP prefixes reported to be acting malicious or used by a DNS changer malware. To find additional malicious resolvers in these prefixes, we scanned each IP in these prefixes and queried for an `A` record for `ad.doubleclick.net`, a popular ad host targeted by malicious resolvers. A resolver is considered malicious if it mis-resolves `ad.doubleclick.net`. Incorrect resolution is determined using the filtering heuristics discussed in Section 3.

Table 4 shows the total number of malicious resolvers we found and the numbers we found reported in online articles. (Details about the owners of these prefixes were derived using the Hurricane Electric BGP Toolkit [5]). A high number of malicious resolvers were found in the IP prefixes in Russia and Ukraine. In contrast, only one resolver was found across the two prefixes we scanned belonging to the U.S. This is likely a result of malicious resolver migration as was reported from prefix 85.255.112.0/24 to 99.198.101.1/19 in 2008 [26].

IP Prefix	Country	Organization	AS	Malicious Resolvers	
				Reported	Found
213.109.0.0/20	Russia	Lipetskie Kabelnie	AS42368	2	166
93.188.160.0/24	Ukraine	Promnet Ltd.	AS36445	1	211
93.188.161.0/24				1	240
93.188.162.0/23				1	422
69.31.52.0/23	US	Pilosoft	AS26627	48	1
99.198.101.0/19		Singlehop	AS32475	51	0

Table 4: Malicious resolvers found in each IP prefix scanned

6.1.1 Behavior seen at .com/.net

Next, we examined the behavior of malicious resolvers in the query traffic seen at Verisign's `.com` and `.net` DNS Top Level Domain (TLD) infrastructure, and its instances of the global DNS root zone. Verisign services over a billion DNS queries per day, hosts well over one hundred million domain names, and because of `.com`'s and `.net`'s popularity, one could argue that almost every heavily used DNS resolver will eventually send queries to this infrastructure [33]. Additionally, Verisign's infrastructure is composed of multiple sites around the world and on several continents. Thus, DNS resolvers that follow the *pinning and polling* behavior described in [33] would likely issue portions of their query load to multiple servers, but ultimately pin themselves to the topologically closest site for the bulk of their queries.

We examined data taken on October 20th, 2011, prior to the take-down of attack infrastructure. It came from roughly half of all of this provider's global sites, which included all of those within the US and several in Europe. Any resolver looking to resolve a host name belonging to `.com` or `.net` would contact these servers in situations where it does not have the host name cached locally. While our view was limited to roughly half of the global sites, it offered us interesting insights into the attack infrastructure. The first observation we made was that none of the known malicious resolvers sent any queries to the TLD servers. This seemed counter-intuitive since we knew that these resolvers returned correct resolutions for most host names queried.

IP	ASN	BGP Prefix	Owner	Websites Hosted	Use
69.31.52.10	AS26627	69.31.52.0/23	Pilosoft	31	Front-end websites
69.31.42.125	AS26627	69.31.40.0/21	Pilosoft	189	
69.31.42.126	AS26627	69.31.40.0/21	Pilosoft	46	
67.29.139.153	AS3356	67.29.136.0/21	Level 3 Communications	147	Fake search engines
63.209.69.109	AS3356	63.209.64.0/21	Level 3 Communications	13	

Table 3: Valid resolutions of malicious websites. The number of front-end websites adds up to 266 but only 263 of these were unique.

When we looked for any queries from the known malicious prefixes, we found 13 IP addresses that were different from the known malicious resolvers. Clearly, the malicious resolvers we found were simply *DNS forwarders*. In fact, many infrastructure providers configure their DNS resolvers to forward their client queries to a more centralized DNS resolver to benefit from caching, better provisioning, etc. Interestingly, 10 of the forwarders were from the Russian prefix and three from the one U.S. prefix. None queried for `ad.doubleclick.net`, the most commonly misdirected ad host. Further, the forwarders queried for several thousand host names during the course of a day but the peak query rate observed from any forwarder was 500 queries per minute, which is lower than one would expect from busy resolvers [33]. Finally, the Russian forwarders were busier and exhibited little in the way of diurnal patterns, possibly indicating the diversity of victims around the globe. However, the forwarders in the US exhibited a distinct peak at 10am.

6.2 Malicious Websites

The ad replacement and click hijacking attacks use front-end websites as a disguise to convince the defrauded ad networks that they serve useful content to Internet users. Additionally, in click hijacking, we observed the use of fake search engines. We found a total of 42 front-end websites and 43 fake search engines during our experiments. In order to expose more malicious websites and explore their features, we devised an iterative algorithm whose basic idea was to take known IP addresses from good resolutions of known malicious websites and find what host names (i.e. reverse look up) they corresponded to using the Hurricane Electric BGP Toolkit [5], which provides a list of host names whose resolutions have returned the IP under consideration. If previously unknown host names were found, we resolved them through malicious resolvers. If they were found to be mis-resolved using the filters we used in Section 3, they were strong candidates for front-end websites. We repeated the same iterative process using the incorrect resolution of the front-end websites and correct resolutions of the fake search engines (since they were never mis-resolved) on the data set of HTTP transactions used in Section 7. This iterative process resulted in a larger set of malicious websites as illustrated in Table 3. Now, we describe their characteristics.

6.2.1 Front-end websites

The iterative process described earlier to identify more front-end websites expanded the list to 263 websites. We manually examined the 42 we found in our experiments. The content on the front-end websites consisted of scrapped content grouped into various categories. In terms of variety of content of these front-end websites, we saw a wide variety. For example, world news (four variations), software news,books for sale, music sites (two variations), video sites (two variations), adult products for sale and others. When these sites offered products for sale, they would eventually link to a legitimate vendor of those products. The front-end websites were all designed to look like web portals with only a few variations in the style sheets for the layouts but with vastly different color schemes. Many of them had a search box that typically only returned sponsored results. They all used one type of "Contact Us" form. Also, the privacy policies listed were fundamentally similar. Despite the dynamic nature of content, the sites hardly made use of client-side scripting, implying that the attacker was deciding on content at its own servers.

The front-end websites used in click hijacking differed from those used in ad replacement in that the latter served ads on their main pages to convince the ad networks they were defrauding that they were indeed serving their ads on those sites. Furthermore, each front-end website had a different sub domain/URL structure specific to the ad networks they were defrauding for management purposes.

The valid resolutions of front-end sites corresponded to three IP addresses belonging to three organizations, as shown in Table 3. Looking at their registration information through *whois* revealed interesting details. The registrant for all but five was "RegName.biz". The rest were registered under "Regtime Ltd". The registration department was either "Advertising network" or the name of the front-end domain followed by "Information department". There were two variants of registrants addresses; one in New York, US while the other was in Benhavn, DK. This homogeneity could have been exploited to find them all.

6.2.2 Fake Search Engines

Even though the fake search engines were not explicitly owned by the attackers, they were used to facilitate the click hijacking attack. Hence, we considered them to be a part of the attack infrastructure. Our iterative process revealed a total of 160 fake search engines.

We manually examined the 43 fake search engines we found during our experimentation and found that 33 of them had only one search text box and search button with links to privacy policy, advertisers, publishers and About Us. They were essentially using one template by Free CSS Templates [3] but with different colors. The rest included news feeds and a weather plugins in addition to the main search text box and button. They were all registered with the same registrar, "Dotster" [2] with two variations of registrant address. The first was in Vancuver, WA while the other was in Encino, CA. Finally, their contact email addresses for inquiries were all of the form "bizdev@fake search engine.com" and "advertise@fake search engine.com". Table 3 shows the two valid IP addresses all the fake search engines resolved to and that both these IP addresses belong to one organization.

6.3 Malicious IP addresses

During the course of our investigations of ad replacement and click hijacking, we found a total of 17 malicious IP addresses where all but two them were used to mis-resolve various ad hosts and search engine host names. The remaining two were *form click* IPs used to simulate form clicks on at-

Attack Type	IP	ASN	BGP Prefix	Owner	Mis-Resolved Valid Hosts/Comments
Ad Replacement	216.180.243.10	AS3595	216.180.224.0/19	Global Net Access, LLC	Ad hosts (ex:"ad.doubleclick.net")
	65.254.36.122		65.254.32.0/20		
	75.102.23.112	AS23352	75.102.0.0/18	Server Central Network	Ad host "pagead2.googlesyndication.com"
	205.234.231.37		205.234.128.0/17		
	205.234.201.230				
Click hijack (visits to search engines)	67.210.15.16	AS36445	67.210.14.0/23	Internet Path	Sponsored results from "googleadservices.com"
	67.210.14.[53-54]				Organic results via script `123.php`
	67.210.15.37 67.210.15.39				Modified search results from "clients1.google.com"
	67.210.15.[70-71]				Modified search results from "ssl.gstatic.com"
Click hijack (visits to regular websites)	75.102.23.111	AS23352	75.102.0.0/18	Server Central Network	Ad replacement or injection of `abc.js` via Google Analytics' script, `ga.js`
	205.234.231.39		205.234.128.0/17		
	205.234.201.229				
Click hijack (form click IPs)	65.60.9.[235-238]	AS32475	65.60.0.0/18	SingleHop, Inc.	Form click IPs
	72.233.76.[66-70]	AS22576	72.233.0.0/17	Layered Technologies	
	94.102.60.6	AS29073	94.102.48.0/20	Ecatel LTD	
Blacklisting defense	67.210.15.38	AS36445	67.210.14.0/23	Internet Path	"safebrowsing.clients.google.com"
Ad replacement & Click hijacking	67.210.14.254				Parked domains
Phishing	67.210.14.189				Sub domains of adult sites (ex:"join.cheerleaderauditions.com")
	216.163.137.61	AS14068	216.163.136.0/23	Playboy	
	62.141.56.61	AS31103	62.141.48.0/20	Keyweb AG	

Table 5: Summary of all malicious IP addresses found

tackers' front-end sites.

Using the data set of HTTP transactions used in Section 7 we searched for host names corresponding to the 17 known malicious IP addresses. If new host names were found, we checked for signatures of ad replacement and click hijacking attacks. Also, we checked if the resolution against malicious resolvers conformed to the characteristics observed in Appendix B.

This process resulted in 30 total malicious IP addresses. Their functionality, along with the ISPs they belonged to, is summarized in Table 5. Interestingly, a few of the malicious IPs carried out functionality not observed in our experiments. One malicious IP was used to mis-resolve a host name for Google's Safe Browsing API, which blacklists phishing and malware domains [4]. The attackers were returning an empty response to bypass blacklisting. Another three were used to run a phishing attack on adult sites as described by Trend Micro [23]. We also found an IP address that was used to mis-resolve parked domains so attackers could divert their traffic to their own ad pages.

Although the attack infrastructure was taken down [27, 34, 36], we found the IP addresses shown in red color in Table 5 to be still up and functioning for a couple of weeks after the take down. Incidentally, all of them belong to one ISP, "Server Central Network".

7 Impact of the Ad Fraud Scheme

To understand the impact of ad replacement on real users, we placed a network monitor on a Broadband Remote Access Server (BRAS). The BRAS we used is an aggregation point for Digital Subscriber Lines (DSLs) for a large Tier 1 ISP's customers located in the United States (U.S.) and serves approximately 17,000 active broadband subscribers. Our analysis was conducted using aggregated data collected from all HTTP traffic transiting this particular BRAS on 2/15/2011. We only had access to HTTP headers and no data packets. The privacy of the subscribers was preserved since the dynamic IP addresses were not mapped to individual households and the study focused on the aggregate traffic across all the subscribers. Further, since query strings in URLs can contain sensitive information, we did not examine the query strings in order to protect subscriber privacy.

Our data contained the source IP, the destination IP, the URL, the response status code, the user agent, and the referrer field for each HTTP request. We used it to reconstruct user sessions, to aid in estimating the occurrence of each type of attack. To find user sessions, we divided the traffic by source IP, which could include multiple users, and then divided each of the resulting sets by the transmitted user-agent string to separate out sessions from the same browser. Each of the user-agent/source IP sets of traffic was then grouped into sessions by identifying time gaps in the traffic. Subsequently, we extracted sessions containing transactions to the known malicious IPs and chained them via the HTTP referrer fields. Chains with malicious IPs were then checked to determine entry points for the attacks. We additionally checked all chains for known entry points in order to find other chains that used unknown malicious IPs.

A summary of the results is in Table 6. *We found 37 infected user lines in our data.* From these lines, we observed 2,811 possibilities for the ad replacement attack, of which 1,800 are successful. Also, we found 144 click hijacking attacks. Of the latter, 111 were organic search result hijacks done by injecting the `abc.js` script through Google Analytics. 6 were hijacks of sponsored search results and the remaining 27 were cases where the attackers changed target URLs for search results via mis-resolution of `clients1.google.com`. We note that there were many more calls to `abc.js` than the successful hijacks reported in Table 6: 2,334. However, these attack chains were not completed. There are several possible explanations to this. First, the attacker may have a strategy to limit the number of click hijacks per machine in a given time-frame, to reduce the footprint of the fraud. Second, it is also possible that there was an undiscovered attack mode. Third, the attacker's infrastructure may have changed between data collection and the mapping of the attack chains. Finally, we note that two of the click hijacking attack modes shown in Figure 3, namely, those that start with the injection of `123.php` on a non search engine website and mis-resolution of `ssl.gstatic.com` were not observed in our data likely because they modes were newly added.

This data also revealed that *257 legitimate content publishers lost revenue due to this fraud scheme.* 17 lost ad revenue for at least 20 ad impressions and clicks each. Also, *we saw 21 different ad hosts whose ad networks lost revenue, with* `ad.doubleclick.net` *being the most popular. They belonged to 20 ad networks.*

While we realize that our data is not a representative of the world or even the United States (U.S.), we extrap-

Total subscriber lines	17,000
Total HTTP requests	55,024,300
Infected lines	37
HTTP requests from infected lines	295,699
Possible ad Replacement attacks	2,811
Actual ad replacement attacks	1,800
Click hijacking attacks	144
Organic search result hijacks	111
Sponsored search result hijacks	6
Hijacks via changed search result targets	27

Table 6: Impact of ad fraud on DSL subscribers of a Tier 1 ISP

olate the numbers to compare our estimations with that of Operation Ghost Click [27, 34, 36]. There are almost 86 million subscription lines in the U.S. [9]. An extrapolation from our data would suggest that 186,574 subscription lines were infected. Extrapolating it world-wide, there would be 1,176,795 infected lines, as there are 540 million subscription lines world wide. Note that there may be multiple computers per subscription line. Member countries of the Organisation for Economic Co-operation and Development (OECD) typically have three times as many Internet users as they do subscription lines, determined by comparing the number of subscription lines according to [9] and the number of Internet users according to [10]. Accounting for this, we estimate the affected users to be 3.53 million, which is somewhat consistent with the estimate of 4 million given by the FBI.

8 Potential Mitigation Strategies

Ads are an important component of the Internet economy. Consequently, while attackers find them to be an attractive revenue stream, ad networks take steps to combat the threat [25, 28]. The ad networks employ various techniques to shield themselves and their advertisers from ad fraud. For example, many audit publisher websites to ensure that they indeed serve the content they claim to be serving. The attackers of the ad fraud scheme discussed in this paper provisioned their front-end websites to render this defense ineffective since the correct resolutions of their front-end websites displayed web pages that looked legitimate. Along the same lines, ad networks also try to match traffic to publisher websites with respect to their page rankings. Further, historical information about publishers is also used to detect sudden changes in ad traffic patterns. We lack data to judge if the front-end websites and fake search engines used by the attackers would be caught under these tests. Next, we discuss possible methods for detecting ghost clicks. Each of these methods rely on the special characteristics of the ad fraud scheme we studied.

- Serving bluff ads: Researchers in [24] explored the use of bluff (bait) ads which are fake ads instrumented to detect illegitimate clicks. They can also be used to detect misdirected human clicks of the types attackers in our ad fraud scheme exploited.

- Finding fake publisher websites: We noted homogeneity among front-end websites and fake search engines. As ad networks discover fraudulent publisher websites, they can leverage homogeneity to find other sister websites.

- Using HTTP with integrity: A simple solution to prevent misdirection is to use a partially secure version of HTTP, namely HTTPI (HTTP with integrity) [13], offers a practical solution advertisers can now employ.

- Monitoring and scrutinizing unexpected DNS resolvers: This defense attacks the key component of the ad fraud we studied – that victims' DNS resolvers were incorrectly configured to attacker-controlled machines. ISPs can monitor DNS requests and responses traversing their organizational borders to find unexpected DNS resolvers. Once found, their responses can be scrutinized using filtering heuristics similar to ones we described in Section 3.

- Identifying accounting discrepancies: Many websites use equivalents of Google Analytics to keep track of the visitors to their sites. Publishers and ad networks can employ better auditing to detect discrepancies.

9 Related Work

Ad fraud has recently been studied. Authors in [31] and [16] reverse engineered clickbots, `Fiesta`, `7cy` and `Clickbot.A` to understand their behavior. Their methodology to understand bot modus operandi is comparable to ours. In a complementary work, Miller et al. [35] investigated a clickbot and used that understanding to study how ad networks can identify and combat click fraud. In the same vein, Dave et al. [17] proposed a methodology advertisers could use to measure click spam rates on their ads. These works differ from ours in that they focus on clickbots.

Zhang et al. conducted a study in which they bought traffic and explicitly located the presence of human clickers as well as clickbots [42]. Their work proves the existence of human clickers in ad fraud while clickbots were well known earlier. Our work expands the landscape of ad fraud to include misdirected human clicks. Since the only malware involved in the fraud we investigated is a DNS changer malware, our work demonstrates the power of DNS misdirections in creating novel ad fraud schemes. Trend Micro researchers helped the FBI in Operation Ghost Click and were investigating the infrastructure we dissect here before us and helped bootstrap our work. Even though their blog provides an account of the takedown and botnet operation [22], to our knowledge, we are the first to document the anatomy of both attacks rigorously.

Inflight modification of Web content has been studied by Zhang et al. in [41]. While the topic of inflight modification may appear orthogonal to ad fraud, the authors discovered that much of the observed modification was attributable to malicious DNS resolvers and ad fraud. These resolvers were sometimes compromised resolvers at the ISPs and sometimes resolvers victims of DNS changer malware were pointed to. This work does not look into how the observed ad fraud was carried out but emphasizes the need to watch DNS resolvers closely.

The malicious resolvers that were central to the ad fraud scheme studied in this paper were open resolvers. In [14], Dagon et al. examined the entire IPv4 space to find open DNS resolvers in the Internet and then to determine when they lied. This is a similar problem to one we encountered while developing heuristics to conclude when malicious resolvers lied. Dagon et al. determined correctness of a resolver response by comparing net blocks of the answers, and then by visiting incorrect resolutions. We found this approach to be problematic in our context. This is because ad hosts typically use CDNs and the answers from geographically displaced CDN servers will legitimately be from different net blocks. Further, ad hosts return non-deterministic ad URLs, ruling out re-visitation. Zdrnja et al. captured DNS replies at the University of Auckland Internet gateway in order to

detect unusual DNS behavior [39]. They focused on typo squatting, domains abused by spammers and fast-flux domains. Our work is different in that we investigate all types of incorrect resolutions whether they were for search engines, ad hosts, parked domains or hosts for antivirus.

The network of victims infected with DNS changer malware can loosely be referred to as a botnet. There have been numerous published reports of botnet takedowns and infiltrations [29, 12]. They typically focus on botnets involved in spam, distributed denial-of-service (DDoS) attacks or phishing. The difference in purpose in our case rules out the need for a traditional botnet command-and-control (C&C).

10 Conclusions

Our work highlights the increasing sophistication of ad fraud schemes. Specifically, the attackers were making creative use of misdirected clicks from real human beings in order to steal CPM and CPC revenue from 20 different ad networks, all the while fooling over a dozen other ad networks that they earned revenue for these clicks and impressions at 423 publisher websites. A critical enabler for the documented attacks was DNS changer malware, which misled victims to resolvers located in Eastern Europe instead of resolvers of their choice, such as those belonging to their ISPs. Our work points to the need to closely monitor DNS resolvers Internet users query.

11 References

[1] Alexa: The web information company. http://www.alexa.com/.

[2] Dotster Inc.: Domain registration and website hosting. http://www.dotster.com/.

[3] Free CSS templates. http://www.freecsstemplates.org/.

[4] Google safe browsing API. http://code.google.com/apis/safebrowsing/.

[5] Hurricane Electric BGP toolkit. http://bgp.he.net/.

[6] The official easylist website. https://easylist.adblockplus.org.

[7] Trend Micro threat encyclopedia. http://threatinfo.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=TROJ5FDNSCHANG12EBM&VSect=P.

[8] Ungefiltert surfen mit freien nameservern. http://www.ungefiltert-surfen.de/.

[9] Total fixed and wireless broadband subscriptions by country. Organization for Economic Co-operation and Development, 2010.

[10] Central Intelligence Agency World Factbook, 2011.

[11] How does Google detect invalid clicks? http://adwords.google.com/support/aw/bin/answer.py?hl=en&answer=6114, 2011.

[12] Cho, C. Y., Caballero, J., Grier, C., Paxson, V., and Song, D. Insights from the inside: A view of botnet management from infiltration. In *USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)* (2010).

[13] Choi, T., and Gouda, M. HTTPI: An HTTP with integrity. In *International Conference on Computer Communications and Networks (ICCCN)* (2011).

[14] Dagon, D., Provos, N., Lee, C. P., and Lee, W. Corrupted DNS resolution paths: The rise of a malicious resolution authority. In *ISOC Network and Distributed Security Symposium (NDSS)* (2008).

[15] Daswani, N., Mysen, C., Rao, V., Weis, S., Gharachorloo, K., and Ghosemajumde, S. *Crimeware: Understanding New Attacks and Defenses*. Addison-Wesley Professional, 2008, ch. Online Advertising Fraud.

[16] Daswani, N., and Stoppelman, M. The anatomy of Clickbot.A. In *USENIX Workshop on Hot Topics in Understanding Botnets* (2007).

[17] Dave, V., Guha, S., and Zhang, Y. Measuring and fingerprinting click-spam in ad networks. In *ACM SIGCOMM* (2012).

[18] Eckman, B. Significant rogue DNS activity to 85.255.112.0/22. http://lists.sans.org/pipermail/unisog/2006-November/026937.html.

[19] F-secure. Trojan:osx/dnschanger. http://www.f-secure.com/v-descs/trojan_osx_DNSchanger.shtml.

[20] Feike Hacquebord. Making a million, part two: The scale of the threat. http://blog.trendmicro.com/?p=27054, 2010.

[21] Florio, E. DNS pharming attacks using rogue DHCP. http://www.symantec.com/connect/blogs/DNS-pharming-attacks-using-rogue-dhcp, December 2008.

[22] Hacquebord, F. Esthost taken down - biggest cybercriminal takedown in history. http://blog.trendmicro.com/esthost-taken-down-biggest-cybercriminal-takedown-in-history, November 2011.

[23] Hacquebord, F., and Lu, C. Rogue domain name system servers part 2. http://blog.trendmicro.com/rogue-domain-name-system-servers-part-2, September 2007.

[24] Haddadi, H. Fighting online click-fraud using bluff ads. In *SIGCOMM Computer Communications Review (CCR)* (2010).

[25] Harvey, S. Invalid clicks and online advertising. http://www.google.com/doubleclick/pdfs/DoubleClick-04-2007-Invalid-Clicks-and-Online-Advertising.pdf, April 2007.

[26] Jose, N. Rogue DNS servers on the move. http://asert.arbornetworks.com/2008/11/rogue-dns-servers-on-the-move/.

[27] Krebs, B. Biggest cybercriminal takedown in history. http://krebsonsecurity.com/2011/11/malware-click-fraud-kingpins-arrested-in-estonia/.

[28] Kshetri, N. The economics of click fraud. In *IEEE Symposium on Security and Privacy* (2010).

[29] Leder, F., Werner, T., and Martini, P. Proactive botnet countermeasures – an offensive approach. In *Cooperative Cyber Defence Centre of Excellence* (2009).

[30] McAfee. New DNSChanger trojan hacks into routers. http://www.trustedsource.org/blog/42/New-DNSChanger-Trojan-hacks-into-routers, June 2008.

[31] Miller, B., Pearce, P., Grier, C., Kreibich, C., and Paxson, V. What's clicking what? Techniques and innovations of today's Clickbots. In *SIG SIDAR Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)* (2011).

[32] Mosuela, L. Search engine redirection malware, how it works (and how to fix it). http://blog.commtouch.com/cafe/malware/how-it-works-search-engine-redirection-malware/.

[33] Osterweil, E., McPherson, D., DiBenedetto, S., Papadopoulos, C., and Massey, D. Behavior of DNS' top talkers, a .com/.net view. In *Passive and Active Measurement Conference (PAM)* (2012).

[34] Sengupta, S. 7 charged in web scam using ads. http://www.nytimes.com/2011/11/10/technology/us-indicts-7-in-online-ad-fraud-scheme.html?_r=2.

[35] Stone-Gross, B., Stevens, R., Zarras, A., Kruegel, C., Kemmerer, R., and Vigna, G. Understanding fraudulent activities in online ad exchanges. In *ACM/USENIX Internet Measurement Conference (IMC)* (2011).

[36] United states district court southern district of New york. Sealed indictment. http://www.wired.com/images_blogs/threatlevel/2011/11/Tsastsin-et-al.-Indictment.pdf, November 2011.

[37] Zdrnja, B. DNS Trojan for the Mac in the wild. http://isc.sans.edu/diary.html?storyid=3595, November 2007.

[38] Zdrnja, B. Rogue DHCP servers. http://isc.sans.edu/diary.html?storyid=5434, December 2008.

[39] Zdrnja, B., Brownlee, N., and Wessels, D. Passive monitoring of DNS anomalies. In *SIG SIDAR Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)* (2007).

[40] Zeltser, L. An overview of the freevideo player Trojan. http://isc.sans.org/diary.html?storyid=1872, 2006.

[41] Zhang, C., Huang, C., Ross, K. W., Maltz, D. A., and Li, J. Inflight modifications of content: Who are the culprits? In *USENIX Large-Scale Exploits and Emerging Threats (LEET)* (2011).

[42] Zhang, Q., Ristenpart, T., Savage, S., and Voelker, G. M. Got traffic? An evaluation of click traffic providers. In *WICOM/AIRWeb Workshop on Web Quality (WebQuality)* (2011).