Financial Lower Bounds of Online Advertising Abuse

A Four Year Case Study of the TDSS/TDL4 Botnet

Yizheng Chen[1], Panagiotis Kintis[1], Manos Antonakakis[2], Yacin Nadji[1], David Dagon[1], Wenke Lee[1], and Michael Farrell[3]

[1] Georgia Institute of Technology, School of Computer Science,
{yzchen,kintis,yacin}@gatech.edu, dagon@m.sudo.sh, wenke.lee@cc.gatech.edu
[2] Georgia Institute of Technology, School of Electrical and Computer Engineering,
manos@gatech.edu
[3] Georgia Institute of Technology, Institute for Internet Security & Privacy,
michael.farrell@iisp.gatech.edu

Abstract. Online advertising is a complex on-line business, which has become the target of abuse. Recent charges filed from the United States Department of Justice against the operators of the DNSChanger botnet stated that the botnet operators stole approximately US$14 million [11,18] over two years. Using monetization tactics similar to DNSChanger, several large botnets (i.e., ZeroAccess and TDSS/TDL4) abuse the ad ecosystem at scale. In order to understand the depth of the financial abuse problem, we need methods that will enable us to passively study large botnets and estimate the lower bounds of their financial abuse. In this paper we present a system, A^2S, which is able to analyze one of the most complex, sophisticated, and long-lived botnets: TDSS/TDL4. Using passive datasets from a large Internet Service Provider in north America, we conservatively estimate lower bounds behind the financial abuse TDSS/TDL4 inflicted on the advertising ecosystem since 2010. Over its lifetime, less than 15% of the botnet's victims caused *at least US$346 million* in damages to advertisers due to impression fraud. TDSS/TDL4 abuse translates to an average US$340 thousand loss per day to advertisers, which is three times the ZeroAccess botnet [27] and more than ten times the DNSChanger botnet [2] estimates of fraud.

1 Introduction

Many researchers have observed a shift in how botnets are monetized [33], away from traditional spam and bank fraud applications, towards advertising oriented abuse [5]. Large botnets such as Kelihos [25] and Asprox [1] have moved to monetization methods that abuse the online ad ecosystem. Unlike other types of abuse, impression and click fraud are "low risk/high reward" for botmasters, given the inherent difficulty in attributing specific advertising events due to the complexity of the ad ecosystem [37].

To date, the evidence about the amount of ad-abuse attributed to modern botnets is sporadic, mainly because of measurement challenges. Studying the monetization components of botnets in a controlled environment (i.e., honeypots, dynamic malware analysis) requires researchers to *actively engage* in the abuse, which poses ethical challenges. In addition, dynamic malware analysis methods often fall short as botnets move their monetization components away from binaries [20,36], and instead deliver them as separate, non-executable add-on modules. Such drawbacks point to the need for an efficient passive analysis system that can estimate the long-term *monetization campaign* separately from the traditional infection, command and control (C&C) and malware update methods.

To enable efficient, independent, and passive analysis of the long-term ad-abuse caused by botnets, we introduce a novel Ad-abuse Analysis System (A^2S). A^2S leverages spectral clustering methods on passive DNS datasets to identify the network infrastructure (domain names and IP addresses) the botnet under examination uses to perform ad-abuse. It also employs sinkhole datasets to estimate lower bounds of financial loss caused by the botnet's past DNS activities. This technique can estimate financial loss for any botnet where the monetization channel can be mapped to DNS requests. To demonstrate this we analyze a specific botnet's fiscal damage to the advertising world.

Using four years of network datasets, we use A^2S to estimate the scale of the ad-abuse potentially inflicted to advertisers from one of the most notorious botnets in history — TDSS/TDL4. Our conservative estimation shows that TDSS/TDL4 caused financial damage of *at least* US$346 million, or US$340 thousand per day. This estimate was made using less than 15% of the botnet's population, which suggests that the global lower bound describing the financial damages towards the advertisers is likely to be higher.

While these numbers may appear large, they remain an underestimation of the true abuse due to the choices in our measurement methodology. We must emphasize that at every step of our analysis, we err on the side of being overly conservative, as we are interested in lower bounds. This will help us establish an as conservative of a lower bound as possible, using aggressive, empirically driven filtering and relying on the lowest possible estimates for constants used in our financial abuse calculation. We intentionally exclude highly likely TDSS/TDL4 domains in exchange for a safer lower bound estimate.

Our contributions in this paper include:

- An Ad-abuse Analysis System (A^2S) that enables researchers to independently and passively analyze the ad-abuse a botnet inflicts to advertisers. The goal of A^2S is to estimate lower bounds of the advertisers' financial loss caused by the botnet using data-driven approaches. With this knowledge, network operators, such as large ISPs, can design network policies to reduce both (1) the economic gains for adversaries that monetize ads and (2) the overall impact a botnet may have to the online ad ecosystem and the advertisers.
- We use A^2S to study the ad-abuse component of TDSS/TDL4, one of the most complex, sophisticated, and long-lived botnets in the history of the Internet. Using four years of network datasets from one of the largest Internet Service Providers (ISPs) in North America, we study: (1) the network infrastructure necessary to support the ad-abuse operation and (2) the financial model to estimate abuse inflicted by the botnet on advertisers. Our major findings include:
    - Online advertisers lost at least US$346 million to TDSS/TDL4. This amount is based solely on actions by less than 15% of the botnet population. This translates to more than US$340 thousand per day on average, and the abuse was mostly accomplished by impression fraud. It is worth noting that daily abuse levels are three times of recent results reported for ZeroAccess botnet [27] and as large as ten times of the short-lived DNSChanger [2] botnet.
    - With respect to the infrastructure that supported this botnet operation, adversaries employed a similar level of network agility to achieve monetization as they do with traditional botnet C&C communication. *At least* 228 IP addresses

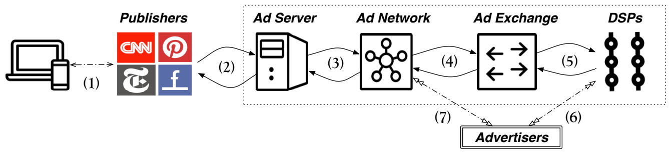

Fig. 1: An overview of the advertising ecosystem.

and 863 domain names were used to support the entire ad-abuse operation over four years. The domain names are available at authors' homepages [3].

2 Background

2.1 The Ad Ecosystem

Figure 1 shows a conceptual view of the *overall* online advertising ecosystem. In general, when a user visits a website (Step (1)), a JavaScript or IFrame dynamically inserts ads. The HTTP session requesting an ad is called an "impression", and the content is sourced at Step (2) via an *ad server*. Ad servers typically work with an *ad network* to serve the impression (Step (3)) and log traffic source for payment. The ad networks are increasingly operated as free services to attract the "long tail" of content owners, but are otherwise monetized through CPM charges (*Cost Per Mille*, i.e., cost per 1,000 impressions) for undifferentiated impressions.

Publishers who source their ads from a search ad network can choose to *syndicate* the ads to other publishers. Search ad networks usually allow syndication in order to reach a wider audience who do not use their own search engines. Thus, there can be several redirections among publishers before Step (2) happens.

Some advertisers work directly with the ad network (Step (7)). However, if a given impression cannot be fulfilled, it is sent to an *ad exchange* (Step (4)). The ad exchange provides market clearance for serving impressions, typically on an individual basis. Other advertisers work with demand-side platform providers (DSPs) to "broker" real-time bidding on impressions through ad exchanges (Step (5) and (6)). DSPs determine how much to bid, based on user-centric features such as IP addresses, cookies, referrers, etc. Instead of charging on CPM basis, they claim anywhere from 5% to 60% of the revenue spent by the advertiser.

If the displayed ad was clicked, the ad server logs which publisher the click comes from, and redirects the user to the advertiser's page. After the click, the advertiser is then charged based on CPC (*Cost Per Click*). The CPC for each click varies based on keywords, publisher popularity, user's profile, location, etc.

Entities in the ad ecosystem perform fraud detection independently. The technical details are not disclosed in public documents [34,13,17]. As a countermeasure for fraud, ad networks employ smart pricing to normalize CPC (Cost-Per-Click) for publishers based on relative conversion rates [14,13]. Examples of conversions include product news subscription, purchase activity, completing an online survey, etc. If click traffic from a publisher results in a low conversion rate compared to other publishers serving similar ads, the ad network may use smart pricing to reduce the CPC used to calculate payment to that publisher. The drawback of the smart pricing policy is that the conversion data are often considered sentitive information and therefore advertisers typically are not willing to share them with the ad networks. In practice, ad networks take many factors into account that would indicate the probability for a conversion [12].

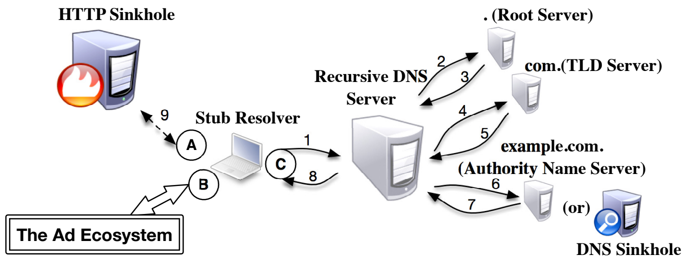

Fig. 2: A high level overview of DNS resolution (1-8), the sinkholing processes (A) and the points where ad-abuse can be observed (B and C).

Nevertheless, since the conversion data are limited, attackers have been able to get positive CPC values even after smart pricing discounts [36].

While smart pricing could reduce the levels of abuse from fraudulent clicks, this is not the case with fraudulent impressions. Only recently, Google and IAB announced the Ad "Viewability" standard in an effort to combat invalid impressions: at least 50% of ad pixels need to be in view for a minimum of one second [15,8]. Advertisers can now choose whether to only bid on *viewable* impressions in the Real Time Bidding process.

2.2 Botnets and Sinkholes

In the Domain Name System (DNS) [23,24], domain names are composed of labels, separated by periods, which correspond to namespaces in a hierarchical tree structure. Each label is a node, and the root label (.) is root of the tree. The hierarchical concatenation of nodes creates a fully qualified domain name. A zone is a collection of nodes that constitute a subtree with *DNS authority servers* responsible for its content. Figure 2 illustrates a typical resolution process. It begins with a stub resolver issuing a domain name resolution request for a domain, `example.com`, to the local recursive DNS server (RDNS) (see step 1, Figure 2). In the event that the RDNS does not have the resolution answer in its cache, it will begin an iterative process to discover it. The RDNS will iteratively "walk" the DNS hierarchy, starting from root server (steps 2 and 3), to the next level of effective top-level domain (TLD) server (steps 4 and 5), and down to the authority name server (ANS) for the requested zone (steps 6). Once the RDNS receives (step 7) the authoritative mapping between the requested domain names and its corresponding answer (e.g., IP address) from the authority, it forwards the answer back to the stub resolver (step 8).

After a command and control (C&C) domain for a botnet is resolved, the next step is a connection attempt (e.g., HTTP GET) from the stub to the C&C server. Network administrators and security researchers often take over such C&C domain names to change their DNS setting, effectively making them point to a new location. This is commonly known as "sinkholing" a domain name [6]. If `example.com` is sinkholed, the stub resolver will establish any future connections to the sinkhole (step 9, Figure 2) rather the adversary's C&C server.

In addition to sinkholing a domain's A/AAAA record, one can also sinkhole the authority name server that serves it. For instance, `example.com` can be sinkholed by changing the ANS to a server under the control of the sinkholing party. Such an action would have the following result: during the DNS lookup chain in Figure 2, after steps 1 to 5, the recursive DNS server will ask the new DNS sinkhole server controlled by the sinkholing party about the authoritative answer for the domain name. Sinkholing both the domain name and the ANS server is a common practice in the security community as it provides telemetry from both the DNS resolution and network communication planes of the threat being sinkholed.

Attackers often change C&C domains to avoid sinkholing. Domain name Generation Algorithms (DGAs) [4,36] can be used to rapidly update the C&C domains to remain agile against sinkholing efforts. A DGA can be implemented client-side in the malware sample itself, or server-side in the C&C server. Intuitively, client-side DGAs can be reverse engineered from the malware sample. Unfortunately, server-side DGAs are much more difficult to understand as reverse engineering requires obtaining the C&C server code, which is often heavily protected by the author. However, monitoring traffic from infected hosts guarantees the observation of C&C domain changes.

2.3 Observing Ad-abuse In Local Networks

To understand where and what an operator can monitor, we need to examine the typical life cycle of a host already infected with malware. First, the malware contacts the C&C server to get its commands. These vary from search engine syndication abuse to traditional impression and click fraud. Next, the malware will attempt to execute the commands by interacting with the ad ecosystem. Stealthy malware carries out these tasks by blending in with users' normal web browsing activities in order to evade detection from anti-abuse components within the ad ecosystem. Additionally, the malware often reports back to the botmaster various byproducts from the monetization activities (e.g., user's search history during the impression or click event) in order to maintain "bookkeeping" for the entire monetization campaign.

Typical egress monitoring functionality can be used to observe different aspects of ad-abuse. Administrators who can inspect the egress of their networks (points A, B and C in Figure 2) are able to independently observe the interactions over DNS and the C&C protocol between the infected hosts, the ad ecosystem, and the ad-abuse infrastructure that supports the particular monetization campaign. From the network's point of view, this observation takes the form of DNS resolutions (i.e., for the domain facilitating ad-abuse from point C in Figure 2) and any application-layer communications between local victims and the ad ecosystem (point B in Figure 2). We select observation points A and C in Figure 2, so we can mine sinkhole and DNS datasets. We should also note that HTTP connections can be observed for the sinkholed domain names (point A in Figure 2). The communications to the sinkhole did not, at any point, reach the ad ecosystem. This means that our efforts to study the botnet did not contribute any additional abuse to the advertisers and other parts of the online advertising ecosystem.

3 Ad-abuse Analysis System

In this section we introduce the Ad-abuse Analysis System (A^2S, Figure 3) that allows administrators to systematically analyze ad-abuse in their networks. The goal of the system is to provide a detailed analysis of the Internet infrastructure that supports ad-abuse. Such information helps administrators to *independently* (1) estimate the level of ad-abuse that victims in the local networks contributed to the entire ad ecosystem and (2) obtain a set of domain names and IPs that can be used for network policy actions. We begin by providing an overview of A^2S.

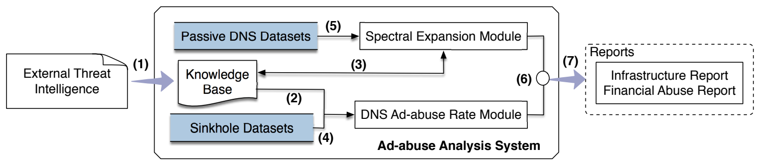

Fig. 3: Overview of the Ad-abuse Analysis System (A^2S).

3.1 System Overview

The input of A^2S is ground truth obtained by either external threat reports or manual analysis of a particular threat (Step (1), Figure 3). These are added to our knowledge base for two modules: the DNS Ad-abuse Rate Module (Step (2)) and the Spectral Expansion Module (Step (3)).

The **Passive DNS** and **Sinkhole** datasets are the input datasets for A^2S. At a high-level, the sinkhole dataset is used to identify the specifics of the command and control communication for monetization purposes and the passive DNS datasets are used to identify the botnet's full infrastructure, and estimate fraud costs at a larger scale. Collecting and handling these datasets are described in more detail in Section 4.

The **DNS Ad-abuse Rate Module** estimates how many ad-abuse events (i.e., C&C connections for impression or click fraud) are typically triggered after a single DNS resolution request for any ad-abuse domain (Step (4)). Multiple ad-abuse actions are often requested by each command received from the C&C server. This can be achieved by "taking-over" a small portion of such ad-abuse domain names for a period of time. Traditional sinkhole methods or commonly used walled garden policy techniques [21] at the recursive DNS level and perimeter egress points of a network can help administrators achieve this goal.

The **Spectral Expansion Module** identifies a set of domain names that have been used by the ad-abuse campaign *historically*. This can be done by combining ground truth from external threat intelligence with large passive DNS datasets (Step (5)). The passive DNS datasets enable the creation of graph between the botnet's victims and the Internet infrastructure that have been contacted by the local botnet victims. Using different sliding temporal windows, the spectral clustering of this graph enables operators to extend the ad-abuse domains to a larger set that is highly related to the ground truth. The module iteratively expands the set of ad-abuse domain names and improves our understanding behind the long-term ad-abuse operation (Step (3)). After expansion, the module sanitizes extended ad-abuse domains using historical WHOIS information in order to eliminate false positives.

The resulting output from both modules will be combined (Step (6)) to derive the final report (Step (7)), which includes all domain names and IP addresses that have been used to facilitate the ad-abuse campaign. The expanded set of ad-abuse domains and their historical DNS lookup volumes are used to approximate a lower bound of financial loss caused by the particular ad-abuse campaign against the ad ecosystem, and in particular the advertisers.

3.2 DNS Ad-abuse Rate Module

The DNS Ad-abuse Rate module quantifies the number of ad-abuse events that are performed after a single DNS request. In this case, the ad-abuse events are the C&C connections issued for impression or click fraud. This allows accurate projection of DNS lookup volumes to the number of total ad-abuse events. To compute the rate,

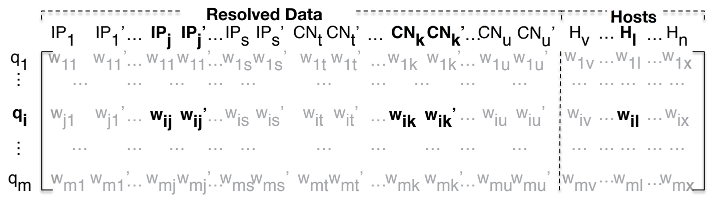

Fig. 4: Association matrix for domain, RDATA, and host.

the module needs to analyze DNS queries and application-layer HTTP requests to sinkholed domains that are part of the ad-abuse campaign.

We define the "DNS Ad-abuse Rate" as $\zeta = y/x$, where x is the number of domain name resolution requests for the sinkholed domains and y is the number of application-layer communication attempts that reflect ad-abuse events. In other words, the module needs to observe x domain name resolution requests and y HTTP connections to the sinkhole, within a time window t, to safely assume a ζ level of ad-abuse happened with each historical ad-abuse domain lookup. Administrators can collect such sinkhole datasets either by acquiring a commercial sinkhole data feed or by independently taking over the ad-abuse domains, locally or globally.

Using ζ, the module can provide the system the ability to *pivot* from "short-term" sinkhole observations to "long-term" passive DNS observations. More specifically, we can project the DNS Ad-abuse Rate over many years of DNS traffic related to the ad-abuse operation using such passive DNS datasets. We now discuss how A^2S mines these datasets.

3.3 Spectral Expansion Module

The Spectral Expansion module uses local network traffic to reason about the domain names used for the ad-abuse operation, over a long time period. The module accurately identifies additional domains based on original ground truth knowledge of the ad-abuse operation, using a large passive DNS dataset. The module derives a larger set of ad-abuse domains, D_A, from the ground truth domains, $D_\$$ using spectral methods on DNS datasets from the local network. The spectral expansion algorithm iterates through the entire DNS query dataset. Each iteration walks over DNS data for a given day, with the ultimate goal of discovering new ad-abuse domains that will be added to the D_A set.

We conservatively assume that unknown ad-abuse domains were queried by a common group of infected hosts, or they pointed to the same Internet infrastructure that served the known ad-abuse domains over the same temporal window. Each day, we create a tripartite graph that "links" candidate domain names, their resolved IP addresses or Canonical Names (CNAMEs), and the network hosts that queried them. The association matrix representing such a graph can be seen in Figure 4. Spectral decomposition of this matrix enables this module to group candidate domain names that either share common Internet infrastructure and/or local network hosts that queried them, via standard clustering methods. Then we analyze the clusters to add domain names to D_A. Domains are added if they have explicit relationships with already known ad-abuse Internet infrastructure or share common infected hosts.

Algorithm 1 formally describes the spectral expansion process. Each iteration of the algorithm processes the DNS resolutions of day, d_i, to update the ad-abuse domain set, D_A. The operator can set δ to determine how the algorithm iterates through time.

Next we discuss the steps in detail for one iteration. Initially we assume that $D_A = D_\$$. The first four steps prepare necessary data for assumbling the association

Algorithm 1 Spectral Expansion Algorithm

```
Require: $\delta$
1: $H \leftarrow \{h | \exists q \in D_A : h$ queried $q$ on day $d_i\}$
2: $D \leftarrow \{q | \exists h \in H : h$ queried $q$ on $d_i\}$
3: $Rdata \leftarrow \{ip | \exists q \in D : q$ resolved to $ip$ historically$\} \cup \{cname | \exists q \in D : q$ resolved to $cname$
   historically$\}$
4: Apply thresholds $\alpha$ and $\beta$ to the sets of $Rdata$ and $H$, respectively, to remove noisy
   IPs and hosts.
5: $M \leftarrow$ relationship between $D$ and $(Rdata, H)$. Normalize by IPs, CNAMEs and Hosts.
6: $S \leftarrow M \times M^T$
7: $U \Sigma V^* \leftarrow SVD(S)$
8: $clusters \leftarrow XMeans(U)$
9: $D_A \leftarrow$ Analyze $clusters$.
10: $i = i + \delta$, Go to line 1.
```

matrix between domains of interest and their resolved answers. In the **first step**, the algorithm identifies all internal network hosts (H) querying any known ad-abuse domain in D_A. In the **second step**, the algorithm narrows down potential unknown ad-abuse domains to all domains (D) queried by infected hosts (H). In the **third step**, we obtain all historical IP addresses and CNAMEs for domain names in D from the local passive DNS database, denoted as *Rdata*.

During the **fourth step**, the algorithm removes any "noisy IP addresses" from *Rdata* and "noisy hosts" from H. IP addresses that are likely used for parking or sinkholing and hosts that are probably a large gateway or part of security research infrastructure can introduce noisy association between domains that do not reflect ad-abuse behavior. The algorithm excludes such "noisy" IPs and internal hosts by using two aggressive thresholds. Note that aggressively removing domains will not affect our lower-bound computation, it will only make our estimates safer.

The first threshold (α) denotes the number of related historical domain names for an IP address from typical network traffic on the local network. We exclude IPs with an unusually high number of domains. The second threshold (β) relates to the cardinality of set D queried by an infected host. In this case, if the number of domains queried by a host is over what we consider as typical for infected hosts in the local network, we exclude it from the set H. The way we reason and select the actual values of α and β will be discussed in Section 5.2.

In the **fifth step**, the algorithm builds an association matrix linking the domains in D with the IP addresses and CNAMEs in *Rdata* and the internal hosts in H that queried them (Figure 4). The rows represent all the domains queried by infected hosts, and the columns reflect historically resolved IPs/CNAMEs and the hosts that queried the domains in the day. We compute two types of weights to assemble the matrix. The first weight reflects the DNS lookup properties from the domains in *Rdata*, with the respect of IPs and CNAMEs. Specifically, the weights w_{ij} and $w_{ij}^{'}$ reflect the timestamp for the first day (w_{ij}) and the last day ($w_{ij}^{'}$) we observed domain name q_i resolving to IP_j. And the weights w_{ik} and $w_{ik}^{'}$ reflect the timestamp for the first and last day that domain name q_i resolved to CNAME CN_k. The second weight reflects a binary indicator of whether the particular domain name in *Rdata* was queried in day d_i by an internal host in H. Specifically, if host $host_l$ queried domain q_i on day d_i, the weight value w_{il} equals 1; otherwise, w_{il} equals 0. After the matrix has been assembled, the algorithm will normalize by row (for each q_i) the sum of "IP" values to one, the sum of "CNAME" values to one, and the sum of "Host" values to one.

In **step six** the algorithm transforms the association matrix $M_{m\times n}$ to its corresponding similarity matrix $S_{m\times m}$. This matrix represents how similar domain name q_i is to any other domain q_j. During the **seventh step**, the algorithm performs Singular Value Decomposition (SVD) on S, and obtains $U\Sigma V^* = SVD(S)$. The first twenty left-singular vectors are kept for **step eight**, which are clustered by XMeans [28].

Step nine analyzes the resulting clusters and finds new ad-abuse domain names. This cluster characterization process propagates the existing labels from ad-abuse domains in our knowledge base to unknown domains. The label propagation rules are based on IP infrastructure overlap and querying host overlap between domains. We discuss how we propagate labels based on cluster specific thresholds in Section 5.2. The known ad-abuse domain names set D_A is updated with the newly discovered domains.

The **tenth** and final step of the algorithm restarts the algorithm from the first step. Depending on the value δ set by the administrator, the algorithm determines the day to check next; for δ equals to 1, the algorithm proceeds to the next day, whereas -1 forces it to go backwards in time. This is very useful when the original ground truth resides in the center of time for our network observations. Taking advantage of the updated set D_A, the system can identify more ad-abuse domains. After reaching the last day of available data according to the iterating direction specified by δ, the algorithm stops.

Finally, the module sanitizes the derived D_A to exclude mistakenly characterized ad-abuse domains. We extract email addresses and name servers from WHOIS for domains in D_A, and compare these with known emails and name servers used for the domains in $D_\$$. If either email or name server matches, the newly discovered domain is kept in D_A. Otherwise, we exclude the domain for financial analysis. Thus, the derived D_A will be used to estimate conservative lower bounds of ad-abuse in the local network.

3.4 Reports On Ad-abuse And Financial Models

Outputs from the DNS Ad-abuse Rate and Spectral Expansion Modules are combined with further analysis of pDNS-DB to generate two reports. The first report describes the network infrastructure used to facilitate the ad-abuse, using historical IP addresses derived from the extended ad-abuse domains D_A. These domains, along with the DNS Ad-abuse Rate and the daily DNS lookup volumes, will help generate the second report that estimates the daily and overall financial impact of ad-abuse to the online advertising ecosystem.

Our financial model to calculate the lower bound of abuse M to the advertisers is:

$$M_{impression} = \sum_i \zeta * R_i * (p_{im} * \frac{\mu_{im}}{1000} * CPM) \tag{1}$$

For each day i, advertisers' loss is calculated based on the number of DNS requests R_i to $d \in D_A$ observed in the local network. $\zeta * R_i$ reflects the total number of ad-abuse HTTP connections for C&C purposes. We consider the connections in $\zeta * R_i$ that result into the p_{clk} component, which reflects the percentage of HTTP connections that corresponds to impression fraud communications. Since each connection may contain multiple impressions, μ_{im} represents the multiplicative factor necessary for the model to derive the total number of impressions. The number of thousand impressions multiplied by the CPM (cost-per-thousand impressions) allows us to calculate the financial loss from the fake impressions.

	Date Range	Size	Records (millions)
DNS Sinkhole	8/1/2012 - 5/31/2013	6.9G	565
HTTP Sinkhole	8/1/2012 - 5/31/2013	248.6G	919
NXDOMAIN	6/27/2010 - 9/15/2014	133.5G	13,557
pDNS-DB	1/1/2011 - 11/6/2014	17.9T	10,209

Table 1: Summary of datasets.

Using model $M_{impression}$ we assume that smart pricing policies were perfect across the entire ecosystem and no click fraud was successful at any point in the lifetime of the botnet operation, whereas the attackers were able to monetize fraudulent impressions from infected hosts. This assumption is realistic since detecting impression fraud has been extremely challenging to date [31,33].

4 Dataset Collection

To increase the situational awareness behind the problem of long-term ad abuse, we chose to analyze the ad-abuse component of the TDSS/TDL4 botnet, which uses a server-side DGA to generate its C&C domains. We describe the collected datasets in this section.

4.1 Sinkhole Datasets

We obtained sinkhole DNS and HTTP traces for the ad-abuse component of TDSS/TDL4 from two security companies. The datasets span over 10 months. All domain names that were sinkholed had a zero time-to-live (TTL) setting, which prevented caching at the recursive DNS server level, forcing it to contact the DNS sinkhole server for every lookup. Moreover, the HTTP sinkhole returned "HTTP 200 OK" answers back to the victims with no content. That is, the sinkhole administrator did not actively engage in ad-abuse.

In order to quantify the DNS Ad-abuse Rate (Section 3.2), we need to understand the type of HTTP connections in the datasets. TDSS/TDL4 employs two C&C protocols to facilitate its ad-abuse operation. Both protocols were present in the HTTP datasets we obtained. The first protocol, "Protocol 1", is the primary mechanism through which the botnet performs impression fraud. This is achieved via an HTTP GET request to the active C&C, which will reply back with a set of advertisement URLs used for impression fraud. Among other information, Protocol 1 also reports the version of the malware behind the infection and a unique identifier for each victim, namely *bid*. All these observations are in-line with data collected and analyzed by other security researchers [26,29]. The second protocol, "Protocol 2", is used to report back information regarding search terms from the victim's browser, the publisher's website where ads have been *replaced* and *clicked on*, and the original ad that was replaced from the publisher's website. A semantically similar behavior of TDSS/TDL4 botnet is identified by Vacha et al. [10], where fraudulent clicks were only generated when a user engaged in real clicks. In order to protect infected users' privacy, the search terms were given to us in an aggregated form such that they cannot be mapped to the individual ID and the infected IP.

In total, we observed 565 million unique DNS resolution requests. 544 million were for Protocol 1 and 21 million were for Protocol 2 connections. This traffic was produced by 47,525 different recursive DNS servers (RDNS) around the world. Hosts with 66,669 unique identifiers (ID) contacted the HTTP sinkhole, using 615,926 different IP addresses. They made 343 million unique `HTTP GET` requests using properly formatted base64 encoded URLs. 919 million connections were recorded, only 0.87%

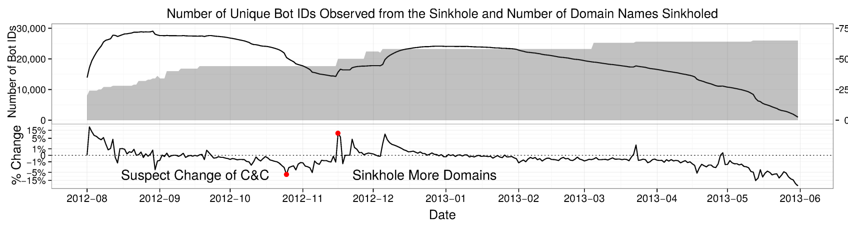

Fig. 5: Top: The line plot shows victim population of the botnet sample that contacted the sinkhole infrastructure, with y-axis on the left. The area plot shows the number of sinkholed domains with y-axis on the right. Bottom: Percent change.

of which reflected Protocol 2 communication, while the rest 99.13% reflected Protocol 1 connections. Thus, we assigned $p_{im} = 99.13\%$ for Equation (1).

4.2 Passive DNS Datasets

We gathered two types of DNS datasets from a large US ISP that represents approximately 30% of DNS traffic in the US. The first is the NXDOMAIN dataset, which covers over four years of DNS queries from clients of the ISP for domains that did not resolve at the time of query. The second dataset we obtained is a historical passive DNS database (pDNS-DB), from the same ISP, containing DNS resource records (RR) [23,24] collected from 1/1/2011 to 11/5/2014.

The queries from the NXDOMAIN dataset are DNS answers with a return code of "NXDOMAIN". The dataset was collected below the recursive DNS servers, effectively capturing the queries from hosts to the recursive DNS servers. Throughout the four-year period, we gained access to 1,295 days of NXDOMAIN data (as the DNS query dataset) from the ISP sensors.

The pDNS-DB dataset contains over 10 billion RRs. Each RR provides resolved data and the daily lookup volume of a queried domain name. The pDNS-DB was collected from 24 geographically diverse ISP collection points in the United States.

5 Analysis and Measurements

In this section, we discuss how we compute the DNS Ad-abuse Rate, and how we propagate ad-abuse domains from ground truth $D_\$$ to the larger set D_A.

5.1 Computing the DNS Ad-abuse Rate

As a sanity check, we measure the average infection duration, the victim population, and the geographic distribution of sinkholed infections.

Figure 6a shows the cumulative distribution function of the *average infection duration* based on IP address and victim ID, a 40-byte long hexadecimal value that was tagged by TDSS/TDL4 malware as *bid* in Protocol 1 communications. The results show a relatively longer infection lifetime for the victims based on the unique identifier than using the victim's IP address. This is reasonable due to the complexity of network egress points, Network Address Translation (NAT) points, and DHCP churn rates [32], as other researchers have already noted.

Second, we measured the victim population coverage of the sinkhole traffic, using the number of unique daily IDs that contacted the sinkhole. Figure 5 illustrates how the number of daily victims changes over time and the percentage of change [16] for the botnet observed from the sinkhole data. In the first two months of the datasets, the number of infected IDs reached a maximum of almost 30,000. After a sudden 6.7% drop in October, the number of IDs seen daily in our datasets decreased, until the middle of November 2012. The decrease indicates that the malware changed C&C domains from sinkholed domains to others. At that point the sinkhole administrators "refreshed" the sinkhole by adding six new domain names for the same botnet. This caused an increase in the number of IDs that were found in the sinkhole datasets. A large number of old IDs reappeared in the sinkhole data after the addition of these six new domains. This observation is expected, as the server side DGA churns through new domains and old infections catch up with the new sinkholed domain names. After a peak of almost 8.9% increase at the end of 2012, the daily victim population remained around 23,000 until the middle of February 2013. Afterwards, the size decreased by a factor of almost 2% daily.

Finally, we examined the geographic distribution of the infected population. As our passive DNS datasets were collected at a US ISP, we want to make sure that the sinkhole dataset contains a reasonable size of victims located in the US. We identified the corresponding CIDR and Autonomous System Number (ASN) for each victim IP address [22], and used historical data from Regional Internet Registries (RIR) to find the country codes for the identified ASNs. Almost half of the sinkhole traffic originates from victims in the US (46.77%). In total, 174 countries were affected, however, only 15,802 infections resided in countries outside the top six countries (US, EU, DE, CA, FR, UK). These results show that TDSS/TDL4 traffic in our pDNS-DB dataset will allow us to study less than 15% of the entire botnet. This is due to the fact that the passive DNS dataset is collected from an ISP in the United States, which represents 30% of the overall DNS traffic in the US.

Computing the DNS Ad-abuse Rate ζ: Since our pDNS-DB dataset was obtained from a US ISP, we calculated the DNS Ad-abuse Rate ζ^{USISP} based on the sinkhole traffic that reflected victims in the particular ISP. This resulted in 9,664 unique victim IDs, 28,779,830 DNS connections, 154,634,443 HTTP Protocol 1 connections and 1,159,027 HTTP Protocol 2 connections over an observation window of 10 months. The mean for the entire ISP as $\zeta^{USISP}_{mean} = 27.62$. which we used as the final DNS Ad-abuse Rate for our experiments. As discussed in Section 4.1, DNS caching will not bias our rate, since the sinkhole administrators had set a TTL equal to zero for the domains they sinkholed.

5.2 Spectral Analysis

We used Algorithm 1 described in Section 3.3 to derive ad-abuse domains set D_A starting from our limited ground truth $D_{\$}$. In this section we discuss the operational challenges we faced while running this algorithm.

Assembling the Association Matrix Before we constructed the association matrix (see Figure 4), we removed noisy IPs and internal hosts from the sets $Rdata$ and H.

Threshold (α) for Noisy IPs: Figure 6b shows the number of historical domain names per IP address, which were manually labeled from the TDSS/TDL4 ad-abuse domains in $D_{\$}$. We observed that under 40% of confirmed TDSS/TDL4 C&C IPs historically have fewer than 1,000 domains pointing to them. The IPs having more historical domains are likely used for parking or sinkholing. We conducted a one-time manual analysis of a set of IP addresses around the limit of 1,000 related historical domains. The analysis revealed that considering IPs with more than 1,000 historical domains as noisy is an aggressive threshold. However, since we are estimating the lower-bound of TDSS/TDL4 ad-abuse operation, falsely removing IPs that were not used for parking or sinkholing will only help our lower bounds goal. That is, such an aggressive threshold will only remove links within the association matrix that would have allowed us to discover additional ad-abuse domains to be added to the set D_A.

Threshold (β) for Noisy Hosts: Figure 6c shows the cumulative distribution of the number of domains queried by infected hosts in a day. Note that the x-axis is in log scale and the y-axis starts at 90%. The plot shows that only 0.7% of infected hosts queried more than 1,000 domain names in a day. These hosts are likely gateways or research infrastructure that cannot link known and unknown ad-abuse domains reliably during the clustering process. Thus, we used the 1,000 mark as threshold. This means that any host that queried more than 1,000 domains in a day was instantly excluded. This should take care any network address translation (NAT) points and complex infrastructure within the ISP. Again, this is an aggressive threshold, which rather forces us to underestimate the infected hosts (and yield again closer to lower bounds).

Using these thresholds, we constructed the sparse matrix, performed Singular Value Decomposition, and extracted the first 20 left-singular vectors, which we used to cluster the domains in the matrix using XMeans [28].

Cluster Analysis After clustering, we labeled ad-abuse domains based on IP infrastructure and infected hosts.

IP Infrastructure: From clusters containing known ad-abuse domains, we label unknown domains as ad-abuse domains if they share the same IP infrastructure.

Internal (Infected) Hosts: Since TDSS/TDL4 uses a server-side DGA, unknown C&C domains can also be nonexistent domains that never resolve. Therefore, we cannot rely solely on infrastructure to derive the set of domains D_A. Our intuition is that, if a NXDOMAIN is queried by a large percentage of known infected hosts, it is likely to be an ad-abuse domain. We use an aggressive filtering process to find such domains based on internal *host overlaps*. The internal *host overlap* was the percentage of the infected hosts that queried the domain names. We used an aggressive cutoff to on keep NXDOMAINs with the *strongest* 5% host overlaps, which is in line with our lower-bound goal.

Correctness of Spectral Expansion Module We bootstrapped the spectral expansion process with 296 TDSS/TDL4 domains recovered from various public resources. After operating Algorithm 1 2,590 times, going over every day of the NXDOMAIN dataset twice, we discovered 838 new TDSS/TDL4 domains. This means that the total number of TDSS/TDL4 domain names in the set D_A was 1,134. Next, the

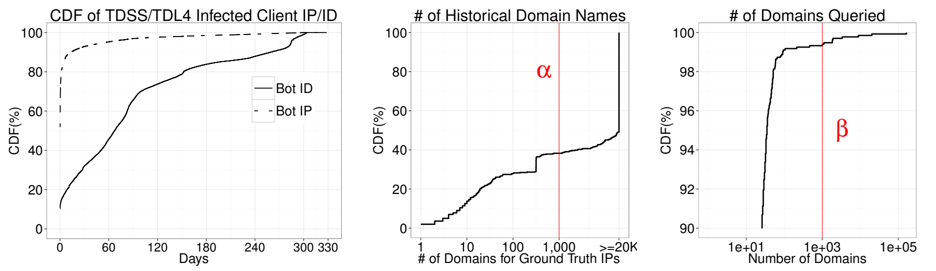

Fig. 6: 6a: Cumulative distribution function (CDF) for the infection duration based on the infection ID and IP address. 6b: CDF for number of related historical domain names per IP from initial ground truth ($D_\$$). 6c: CDF for the number of domains queried by internal hosts (H).

	Detected	Labeled	Lookup Vol. (millions)
Shared Email Address			
email1@nhjhajsukk.cc	216	12	425
email2@aol.com	73	63	205
email3@dikloren.biz	65	9	144
email4@rocketmail.com	112	9	64
email5@kraniccky.com	6	3	57
email6@u7.eu	0	171	261
email7@gmx.com	0	20	28
Shared TDSS Name Server	6	-	4
No Active IP Address			
Sinkholed	64	9	
Two TDSS Parking Services	25	-	
Never Registered	268	-	
Non TDSS/TDL4	3	-	
Total	838	296	

Table 2: Categories of newly detected ad-abuse domains with obfuscated email addresses.

sanitization process reduced D_A to 765 domains based on historical WHOIS (WHOWAS) information from DomainTools. These domains match known TDSS/TDL4 domain registration email addresses or name servers, as shown in Table 2. The lookup volume for these domains will be used for the financial analysis in Section 6.2.

We manually analyzed the rest of the domains, and found that only three domains were mistakenly added to the set D_A by the spectral expansion module, while the rest were related to ad-abuse. The category "No Active IP Address" in Table 2 contains domains that only resolved to known sinkholes, parking IPs, and domains that were never registered. "Sinkholed" represents domains sinkholed by researchers. "Two TDSS Parking Services" refers to domains pointed to the same two parking services used by known TDSS domains during the same time. Lastly, 268 of newly detected domains were never registered. However, based on the large host overlap of these domains with known TDSS domains and name string characteristics, we concluded that these domains were related to the TDSS/TDL4 botnet.

6 Ad-abuse Reports

This section discusses the two reports that summarize the network infrastructure properties behind the ad-abuse component of TDSS/TDL4 and our estimation around financial impact that the botnet brought to the advertisers over four years.

6.1 C&C Infrastructure

Using the 1,131 domains in set D_A, we analyzed the network infrastructure used by the ad-abuse component of the botnet. We separated IP addresses used by these domains into parking, sinkhole, and active categories. Besides well-known parking and sinkholing IPs, we consider IPs with more than 1,000 historical domains to be parking IPs because of the α threshold discussed in Section 5.2. All other IP addresses were considered to be *active*. Figure 7 shows the number of domains resolving into each category over the four year observation period. In total, *at least* 863 domains were registered and the botnet used 228 IP addresses. These IP addresses were used for two years and ten months, until 10/15/2013. These domains were mostly active before the middle of 2012. We should note that during July 2012, a number of researchers started sinkholing some of

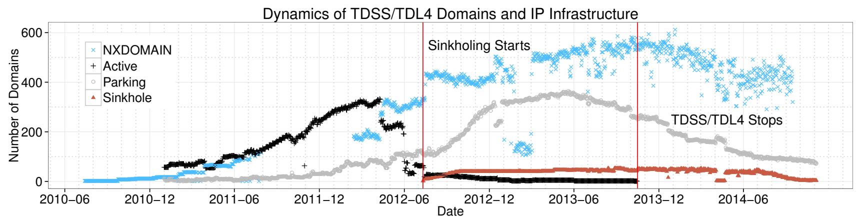

Fig. 7: Evolution of TDSS/TDL4 domains and their IP infrastructure. The number of active domain names daily increased from 2010, and reached the maximum (333) on 4/9/2012. None of the domains resolved to any active IP after 10/15/2013.

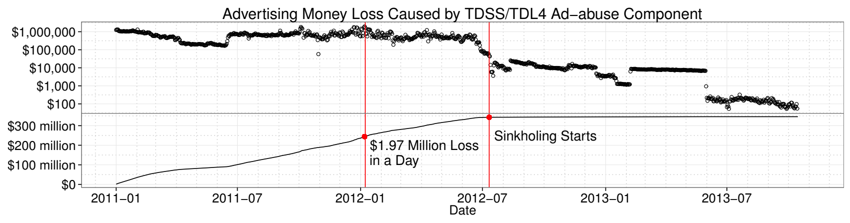

Fig. 8: Top: Daily advertisers' money loss caused by the ad-abuse component of TDSS/TDL4. Bottom: Cumulative financial loss for advertisers. Less than 15% of the botnet population have been involved in ad fraud that cost at least US$346 million.

the TDSS/TDL4 domains. This perhaps forced the botmasters to change monetization tactics as security researchers were investigating the ad-abuse component.

The botnet used a variety of hosting infrastructures to facilitate the abuse. We obtained ASN information for 195 out of 228 total active IP addresses used by the ad-abuse C&C. They are under 49 different Autonomous System Numbers (ASN), 59 CIDRs and 24 countries. Table 3a shows the distribution of the servers around the globe, used by TDSS/TDL4 domains.

6.2 Financial Analysis

We used Equation (1) to estimate the advertisers' financial loss. For our local network (the US ISP) we calculated the DNS Ad-abuse Rate to be $\zeta = 27.62$ (Section 5.1) and the percentage for impression fraud as $p_{im} = 99.13\%$ (Section 4.1). We calculated the daily number of DNS requests R_i to domains used for ad-abuse that resolved to *active* IP addresses. This is an *under-estimation* since we used aggressive thresholds to exclude potentially parked domains in our passive DNS traces (as we discussed in Section 5.2). This resulted in 1.2 billion DNS requests in total. μ_{im} denotes the number of ads returned by each Protocol 1 request (which relates to impression fraud activity). During our analysis, we identified instances where as many as 50 ads were returned from the C&C after each Protocol 1 request. We never saw fewer than 5 ads per request according to network traces of malware execution reported by [26]. Therefore, we used $\mu_{im} = 5$ for our lower bound estimate.

Throughout the lifetime of TDSS/TDL4, we estimate levels of ad-abuse on the order of at least US$346 million using Equation (1). This lower bound is only based on the DNS datasets from the American ISP network to which we had access. Figure 8 shows the distribution of the financial loss caused by TDSS/TDL4 to advertisers. The

Country	IP Addresses	%
RU	42	18.42
US	34	14.91
LV	20	8.77
PT	19	8.33
DE	18	7.90
EU	17	7.46
NL	13	5.70
Other (17)	32	14.04
Unknown	33	14.47
Total	228	100.00

(a) Infrastructure Location

Stakeholders		**Money (millions)**
Advertisers' Capital		346.00
DSP	45%	155.70
Ad Exchange (inbound)	8%	27.68
Ad Exchange (outbound)	8%	27.68
Ad Networks	32%	110.72
Ad Server/Publisher (Affiliates)	**7%**	**24.22**

(b) Financial Break Down

Table 3: 3a: The top 8 countries where C&C infrastructure has been identified. They count towards 71% of the IP addresses. 3b: Financial break down approximation among the entities of the online ad ecosystem, in millions of dollars.

daily financial loss is shown at the top of the figure, and the cumulative financial loss is at the bottom. We observed 1,018 days of active ad-abuse C&C DNS communications, caused by victims in the American ISP. This resulted to an average of US$340 thousand lost daily loss for advertisers. However, before the first sinkholed domain was registered on 7/11/2012, the daily estimate was on average US$616 thousand and peaked to US$1.97 million, on 1/7/2012. After the sinkholing action, the financial impact to the advertisers drastically decreased as the plateau of the bottom plot in Figure 8 shows.

We strongly believe that other networks in the world were affected by this threat based on our sinkhole analysis described in Section 5.1. The victims in the *entire* ISP roughly accounted for 30% of the total botnet population in the US. The infected hosts in the US were less than 50% of the entire botnet population in the world. Thus, our lower bounds may only conservatively estimate loss caused by less than 15% of the entire botnet population.

Cost for Operating the TDSS/TDL4 Infrastructure: The ad-abuse hosting infrastructure was located in 228 different IPs. Without knowing the hosting plans actually used by the botmasters, we have to consider an average cost plan for each service provider to approximate the cost of running the TDSS/TDL4 botnet. Using manual analysis, we conclude that the average minimum (i.e., the botmaster is using the least expensive plan) cost is approximately US$33.62 per month, whereas the average maximum cost is almost US$444 per month. We assume infrastructure is used around the clock. For IPs that we could not link to a particular AS, we assume a flat rate. This rate corresponds to the median of the observed prices around the world. Using this information, we conclude the cost to operate the TDSS/TDL4 C&C infrastructure to be between US$44,000 and US$260,000 over four years.

Potential Financial Reward for the Botnet Operators/ Affiliates: While it is impossible to know for sure what the exact reward may have been, we tried to approximate the revenue that went to the affiliate TDSS/TDL4 entities. To derive the stakeholder and the break-down described in Table 3b we consulted the Chief Technology Officer (CTO) of a large Demand Service Platform company. According to his expert opinion, these are the most typical breakdowns to various entities in the ad ecosystem. As we can see from Table 3b, the potential financial reward for the affiliates is in the order of tens of millions of dollars.

The botmasters and affiliates are likely getting paid as publishers or traffic resellers. In this role, the estimated revenue is 7% of money spent by advertisers, US$24.22 million. Our estimates are in-line with investigations from law enforcement on the amount stolen by fraudulent advertisement campaigns [11,18]. For example, law enforcement agencies recently estimated a minimum level of financial gains on the order of US$14 million dollars for the botmasters behind the DNSChanger botnet [35]. Note that DNSChanger was a significantly smaller botnet that operated over less than half the time period that TDSS/TDL4 was active.

7 Discussion

Our study aims to increase the situation awareness behind botnets that employ sophisticated techniques to abuse the online ad ecosystem and hopefully motivate further research in the space of ad-abuse. In this section we will discuss the most important challenges we faced while analyzing TDSS/TDL4.

7.1 Ground Truth Behind the Financial Loss

The botnets that interact with and monetize the ad ecosystem typically do not target a single entity (i.e., Google, Facebook, or Microsoft etc.). Due to the secrecy within the ecosystem, it is very hard to gather all the datasets from different entities necessary to verify whether the abuse levels we estimated are actually what the advertisers lost. For example, however unlikely it may be, we cannot exclude the possibility that some percentage of the impression fraud could have been detected and stopped by some entities in the ad ecosystem. Unfortunately, we cannot determine how much impression fraud, if any, was blocked, nor by whom. Thus, we had to rely on our own assumptions to estimate the lower bound. However, even in the scenario where one entity had perfect defenses, we cannot reliably assume it to be true for all the other entities in the ad ecosystem. While we contacted several entities in the ad ecosystem, they remain secretive about the methodology and tools that they use to detect fraud. Even if a small percentage (i.e., 30%) of the reported fraudulent traffic evades detection, the losses are still significant.

7.2 Ground Truth Behind TDSS/TDL4

Our goal was to get ground truth around the way the TDSS/TDL4 botnet operates in the wild without contributing to online abuse. To that extent, we decided to gather the ground truth from external reports, and also from analyzing the sinkholing datasets of DGA domain names that supported the monetization module in TDSS/TDL4. Observation of DNS Ad-abuse Rate was made passively from actual infected hosts around the world. The TDSS/TDL4 victims were notified behind this sinkholing operation, and the sinkhole data were released to the operational community and several entities in the online ad ecosystem.

7.3 Smart Pricing Data for Impressions and Clicks

We assumed that perfect smart pricing for CPC was successfully used across the ad ecosystem, whereas all fraudulent impressions impacted the advertisers. However, attackers most likely can still profit from fraudulent clicks after smart pricing. For instance, recent work shows the actual CPC charged after smart pricing was between 10 to 30 cents for ZeroAccess [27]. Smart pricing is hard since not all conversion rates can be effectively measured. Not all conversion actions were logged and shared between advertisers and ad networks/exchanges. The fact that TDSS/TDL4 does both impression and click fraud implies that the monetization technique tried to avoid detection by generating positive click-through rates.

We chose to account for all the impressions since we do not have knowledge about how impression fraud was actually handled by the ad networks and ad exchanges. Although new standard of Ad Viewability has been announced and deployed to prevent advertisers from spending money on invalid ad impressions [15,8]. However, since there is almost no documentation about how impression fraud was dealt with by ad networks and ad exchanges when TDSS/TDL4 was active (before October 2013), it is not unreasonable to assume that a significant portion (if not all) of the impressions most likely went undetected.

8 Related Work

Operating a sinkhole is a safe, passive way to collect data regarding network connections between malware and the servers they try to contact. Malware needs to find a way to contact its Command and Control (C&C) server [7], which cannot always be done through P2P protocols, since network operators often block them. In the case of TDSS/TDL4, the malware uses P2P as an alternative communication method [30]. Data collected from a sinkhole operation can be used to measure the network behavior of a botnet. For example, [32] used sinkhole to uniquely identify infected hosts.

Studies on ad abuse often focus on the ad network's perspective [19,10]. Daswani et al. [9] showed how the value chain of ad-abuse operates online through the "Clickbot.A" botnet of 100,000 hosts. Springborn et al. [31] studied pay-per-view networks and described how millions of dollars are lost by fraudulent impressions annually. Moreover, Stone-Gross et al. [33], studied abuse from both a botnet's and ad network's point of view, showing the large amount of money the botnet can make. These works carefully focus on specific parts of the ad ecosystem, while ours characterizes overall abuse impact by using edge-based metrics.

The work most similar to ours is the recent ZeroAccess study [27] that estimated daily advertising losses caused by the botnet by analyzing one week of click fraud activities during a takedown against the ad-abuse component of ZeroAccess, mainly from the view of a single ad network. While the ZeroAccess study was novel, it did not help large network administrators independently measure the levels of ad-abuse originating from their network environments. Our system addresses these limitations from previous studies by studying the ad-abuse problem passively at the edge of the Internet over a multi-year time period.

9 Conclusion

This study aims to quantify the scale of online advertising abuse. To achieve this we present a novel system, Ad-abuse Analysis System (A^2S), able to conservatively estimate the long-term damage of the monetization component botnets use against the ad ecosystem. Using A^2S we studied one of the most notorious botnets that fraudulently monetized the ad ecosystem for four years: TDSS/TDL4. Using passive DNS and sinkhole observations, we were able to estimate TDSS/TDL4's lower bounds for its ad-abuse: no more than 15% of botnet victims were responsible for at least US$346 million in financial loss to online advertisers since 2010. This includes a peak average daily loss of almost US$2 million at the height of the botnet's ad-abuse activity in early 2012. Overall, these figures reveal the extent of the abuse that botnets could bring to the advertisers over time, making ad-abuse a low risk and high reward monetization method for botmasters. The estimated lower bounds suggests the importance of additional research efforts in the ways botnets are being monetized.

Acknowledgements. The authors would like to thank Dr. Brett Stone-Gross and Dag Liodden for their comments and feedback. This material is based upon work supported in part by the US Department of Commerce under grant no. 2106DEK and Georgia Tech Research Institute (GTRI) IRAD grant no. 21043091. 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 US Department of Commerce nor GTRI.

References

1. Click-Fraud Attacks Being Used to Deliver More Sinister Threats. `http://www.tripwire.com/state-of-security/security-data-protection/cyber-security/click-fraud-attacks-being-used-to-deliver-more-sinister-threats/`
2. DNS Changer Remediation Study. `https://www.m3aawg.org/sites/default/files/document/GeorgiaTech_DNSChanger_Study-2013-02-19.pdf`
3. TDSS/TDL4 Domain Names. `http://www.cc.gatech.edu/~ychen462/files/misc/tdssdomains.pdf`
4. Antonakakis, M., Demar, J., Stevens, K., Dagon, D.: Unveiling the Network Criminal Infrastructure of TDSS/TDL4 DGAv14: A case study on a new TDSS/TDL4 variant. Technical Report, Damballa Inc.,Georgia Institute of Technology (GTISC) (2012)
5. Blizard, T., Livic, N.: Click-fraud monetizing malware: A survey and case study. In: Malicious and Unwanted Software (MALWARE), 2012 7th International Conference on. pp. 67–72. IEEE (2012)
6. Bruneau, G.: DNS sinkhole (2010), `http://www.sans.org/reading_room/whitepapers/dns/dns-sinkhole_33523`
7. Bruneau, G., Wanner, R.: DNS Sinkhole. Tech. rep., SANS Institute InfoSec Reading Room (August 2010), `http://www.sans.org/reading-room/whitepapers/dns/dns-sinkhole-33523`
8. Bureau, I.A.: Viewability Has Arrived: What You Need To Know To See Through This Sea Change. `http://www.iab.net/iablog/2014/03/viewability-has-arrived-what-you-need-to-know-to-see-through-this-sea-change.html` (2014)
9. Daswani, N., Stoppelman, M.: The anatomy of Clickbot.A. In: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets. pp. 11–11. USENIX Association (2007)
10. Dave, V., Guha, S., Zhang, Y.: Measuring and fingerprinting click-spam in ad networks. In: Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication. pp. 175–186. ACM (2012)
11. FBI New York Field Office: Defendant Charged In Massive Internet Fraud Scheme Extradited From Estonia Appeared In Manhattan Federal Court. `http://tinyurl.com/7mfrtqs` (April 2012)
12. Google: About smart pricing. `https://support.google.com/adwords/answer/2604607?hl=en`
13. Google: Ad traffic quality resource center. `http://www.google.com/ads/adtrafficquality/`
14. Google: How google uses conversion data. `https://support.google.com/adwords/answer/93148?hl=en`
15. Google: Just in time for the holidays — viewability across the google display network. `http://adwords.blogspot.co.uk/2013/12/just-in-time-for-holidays-viewability.html` (December 2013)
16. Hyndman, R.J.: Transforming data with zeros. `http://robjhyndman.com/hyndsight/transformations/` (2010)
17. Kelleher, T.: How microsoft advertising helps protect advertisers from invalid traffic. `http://advertise.bingads.microsoft.com/en-us/blog/26235/how-microsoft-advertising-helps-protect-advertisers-from-invalid-traffic`

18. LawFuel Editors: Massive Internet Fraud Nets Extradicted Estonian Defendant at Least $14 Million. http://www.lawfuel.com/massive-internet-fraud-nets-extradicted-estonian-defendant-least-14-million (October 2014)
19. Li, Z., Zhang, K., Xie, Y., Yu, F., Wang, X.: Knowing your enemy: understanding and detecting malicious web advertising. In: Proceedings of the 2012 ACM conference on Computer and Communications Security. pp. 674–686. ACM (2012)
20. Matrosov, A.: TDSS part 1 through 4. http://resources.infosecinstitute.com/tdss4-part-1/ (2011)
21. Messaging Anti-Abuse Working Group and others: MAAWG Best Practices for the use of a Walled Garden. San Francisco, CA (2007)
22. Meyer, D., et al.: University of oregon route views project (2005)
23. Mockapetris, P.: Domain names - concepts and facilities. http://www.ietf.org/rfc/rfc1034.txt (1987)
24. Mockapetris, P.: Domain names - implementation and specification. http://www.ietf.org/rfc/rfc1035.txt (1987)
25. Neville, A.: Waledac reloaded: Trojan.rloader.b. http://www.symantec.com/connect/blogs/waledac-reloaded-trojanrloaderb (2013)
26. Parkour, M.: Collection of pcap files from malware analysis. http://contagiodump.blogspot.com/2013/04/collection-of-pcap-files-from-malware.html (2013)
27. Pearce, P., Dave, V., Grier, C., Levchenko, K., Guha, S., McCoy, D., Paxson, V., Savage, S., Voelker, G.M.: Characterizing large-scale click fraud in zeroaccess. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. pp. 141–152. CCS '14, ACM, New York, NY, USA (2014), http://doi.acm.org/10.1145/2660267.2660369
28. Pelleg, D., Moore, A.W.: X-means: Extending k-means with efficient estimation of the number of clusters. In: Proceedings of the Seventeenth International Conference on Machine Learning. pp. 727–734. ICML '00, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2000), http://dl.acm.org/citation.cfm?id=645529.657808
29. Rodionov, E., Matrosov, A.: The evolution of tdl: Conquering x64. ESET, June (2011)
30. Rossow, C., Andriesse, D., Werner, T., Stone-Gross, B., Plohmann, D., Dietrich, C.J., Bos, H.: Sok: P2pwned-modeling and evaluating the resilience of peer-to-peer botnets. In: Security and Privacy (SP), 2013 IEEE Symposium on. pp. 97–111. IEEE (2013)
31. Springborn, K., Barford, P.: Impression fraud in online advertising via pay-per-view networks. In: Proceedings of the 22nd USENIX Security Symposium (Washington, DC). Citeseer (2013)
32. Stone-Gross, B., Cova, M., Cavallaro, L., Gilbert, B., Szydlowski, M., Kemmerer, R., Kruegel, C., Vigna, G.: Your botnet is my botnet: analysis of a botnet takeover. In: Proceedings of the 16th ACM conference on Computer and communications security. pp. 635–647. ACM (2009)
33. Stone-Gross, B., Stevens, R., Zarras, A., Kemmerer, R., Kruegel, C., Vigna, G.: Understanding fraudulent activities in online ad exchanges. In: Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference. pp. 279–294. ACM (2011)
34. Tuzhilin, A.: The lanes gifts v. google report. Official Google Blog: Findings on invalid clicks, posted pp. 1–47 (2006)
35. United States District Court: Sealed Indictment. http://www.wired.com/images_blogs/threatlevel/2011/11/Tsastsin-et-al.-Indictment.pdf (October 2011)
36. Wyke, J.: ZeroAccess. http://sophosnews.files.wordpress.com/2012/04/zeroaccess2.pdf (2012)
37. Zhang, Q., Ristenpart, T., Savage, S., Voelker, G.M.: Got traffic?: an evaluation of click traffic providers. In: Proceedings of the 2011 Joint WICOW/AIRWeb Workshop on Web Quality. pp. 19–26. ACM (2011)