Knowing Your Enemy: Understanding and Detecting Malicious Web Advertising

Zhou Li*
Indiana University at Bloomington
lizho@indiana.edu

Kehuan Zhang†
Indiana University at Bloomington
kehzhang@indiana.edu

Yinglian Xie
MSR Silicon Valley
yxie@microsoft.com

Fang Yu
MSR Silicon Valley
fangyu@microsoft.com

XiaoFeng Wang
Indiana University at Bloomington
xw7@indiana.edu

ABSTRACT

With the Internet becoming the dominant channel for marketing and promotion, online advertisements are also increasingly used for illegal purposes such as propagating malware, scamming, click frauds, etc. To understand the gravity of these malicious advertising activities, which we call *malvertising*, we perform a large-scale study through analyzing ad-related Web traces crawled over a three-month period. Our study reveals the rampancy of malvertising: hundreds of top ranking Web sites fell victims and leading ad networks such as DoubleClick were infiltrated.

To mitigate this threat, we identify prominent features from malicious advertising nodes and their related content delivery paths, and leverage them to build a new detection system called *MadTracer*. MadTracer automatically generates detection rules and utilizes them to inspect advertisement delivery processes and detect malvertising activities. Our evaluation shows that MadTracer was capable of capturing a large number of malvertising cases, 15 times as many as Google Safe Browsing and Microsoft Forefront did together, at a low false detection rate. It also detected new attacks, including a type of click-fraud attack that has never been reported before.

Categories and Subject Descriptors

H.3.5 [**[Information Storage and Retrieval**]: Online Information Services Web-based services

Keywords

Online Advertising, Malvertising, Statistical Learning

1. INTRODUCTION

Visiting any commercial Web site today, rarely will you not bump into banner advertisements (*ads* for short). Such Web advertising has already grown into billion-dollar businesses [36]. Compared to traditional media, online advertising is more convenient and economic. One can easily set up an account with major advertisers such as DoubleClick, and immediately push her marketing messages to a large population. Unfortunately, this bless can also turn into a curse: hackers and con artists have found Web ads to be a low-cost and highly-effective means to conduct malicious and fraudulent activities. In this paper, we broadly refer to such ad-related malicious activities as *malvertising*, which can happen to any link on an ad-delivery chain, including publishers, *advertising networks* (*ad network*), and advertisers. A well-known example is New York Times' malvertising incident, in which a fake virus scanner was found on its home page [32]. Indeed, malvertising becomes a vibrant underground business today, endangering even those who trust only the contents from reputable Web sites.

Anti-malvertising. Both industry and academia have been working on this threat, typically through inspecting ads to detect their malicious content [22]. However, malicious ads often use obfuscation and code packing techniques to evade detection. Further complicating the situation is the pervasiveness of *ad syndication*, a business model in which an ad network sells and resells the spaces it acquires from publishers to other ad networks and advertisers. Ad syndication significantly increases the chance of posting malicious content on a big publisher's Web site. It allows a malicious ad network to deliver ads directly to a user's browser, without the need of submitting them through the more reputable ad networks and publishers from whom it gets the ad space. Furthermore, attackers continue to invent new, stealthy strategies for exploiting ad-delivery channels: a prominent example is leveraging a compromised publisher page to hijack user traffic into clicks (Appendix C).

Thus, despite years' effort, anti-malvertising remains challenging with many open questions. Particularly, little is known about the infrastructure used to deliver malicious ad content. One may ask: how do attackers get onto the ad networks? what roles do malicious nodes [1] play in a malvertising campaign? how do they hide their activities from detection? An in-depth understanding of these issues can help identify the weakest link in the malvertising infrastructure, and present a new angle to address them using information that characterizes not just individual entities, but their roles and interactions with each other.

Our new findings. In this paper, we report an extensive study on the malvertising infrastructure, based upon a crawling of 90,000 leading Web sites over a 3-month span. Using the Web traffic traces collected through the crawling, we perform a fine-grained, in-depth

*Part of the work was done during Zhou Li's intern at Microsoft Research.

†Kehuan Zhang is also affiliated with The Chinese University of Hong Kong.

CCS'12, October 16–18, 2012, Raleigh, North Carolina, USA.

[1]Here a node represents an entity (e.g., publisher, ad network, advertiser, etc.) on an ad-delivery chain.

analysis on the malvertising cases reported by Google Safe Browsing and Microsoft Forefront, and make the following discoveries:

• *Malvertising scale:* Not only does malvertising infect top Web sites, it also infiltrates leading ad networks like DoubleClick.

• *Evading strategies:* Different cloaking techniques are deployed over malvertising nodes, which work together to evade detection.

• *Properties of malicious parties:* Malicious parties exhibit distinctive features, including their ad-related roles, domain and URL properties, the popularity and the lifetimes of their URLs, and their pairing relations. These features, when viewed in isolation, are often not reliable enough for detection. But when they are viewed collectively in the context of ad delivery infrastructure, they offer a good characterization of malvertising activities.

• *Ad delivery topology:* A malvertising path usually involves multiple malicious domains and they tend to stand close to each other in distance. This observation reveals the topological connection among these malicious parties in the ad context, which can be leveraged to characterize their malicious behaviors (Section 5).

New techniques. The dynamic interactions among malvertising entities and their distinctive features present unique opportunities for detection. As a first step, we model ad-delivery topologies using a simple representation in terms of short path segments that describe the redirection relations among domains. Previous work has also measured the redirection chains of malicious Web activities [27]. However, little has been done to explore such topology information for detection. As malicious nodes often stay close along redirection chains, the use of short ad path segments, combined with node features, effectively leverages this observation as well as other properties specific in the ad context. For example, it is unusual to see multiple consecutive domains irrelevant to ads along an ad-delivery path and our representation naturally captures such suspicious cases. Since this approach does not depend on Web page content, it is robust to code obfuscation. Further, it is fundamentally difficult for attackers to alter the features and the interconnect relations of multiple ad entities, especially when some of them are controlled by legitimate domains.

Based on the new representation, we design and implement *MadTracer*, the first infrastructure-based malvertising detection system. We utilize a machine learning framework to automatically generate detection rules on three-node path segments annotated with node attributes. Applying our system to the crawled data from Jun to Oct, 2011, we detect 9568 malvertising redirection chains, each of which involves a unique domain sequences (called *domain-path*, see Section 3.2). Compared to what are detected by Safe Browsing and Forefront combined, our system increases detection coverage by 15 times. Over 95% of the detected malvertising cases have been confirmed so far, either through our collaboration with Microsoft Forefront or by manual validation. Apart from drive-by downloads and fake-AV scams, our system also discovers a new type of click fraud attacks, in which attackers compromise Web sites and hijack normal user traffic into fraudulent ad clicks.

Roadmap. The rest of the paper is organized as follows: Section 2 provides the necessary background information and presents a case study; Section 3 describes the datasets and the terminologies we use; Section 4 elaborates our measurement study; Section 5 describes our new detection techniques; Section 6 reports our experimental results; Section 7 compares our work with related prior research; Section 8 discusses deployment scenarios and future work; Section 9 concludes the whole paper.

2. BACKGROUND

2.1 Online Advertising

Our research focuses on *display ads*, whose contents are loaded automatically to a Web page without the need of user clicks. Display ads are extremely popular, appearing on most highly-ranked Web pages today. Here we describe how this type of ads work.

Actors in Web advertising. Display ads are delivered through a Web-based infrastructure that involves the following major parties:

• *Publishers* display ads on their Web pages on behalf of advertisers. They usually make profit by either *pay-per-impression*, i.e., paid by the number of user views, or *pay-per-click*, i.e., paid by the number of ad-clicks.

• *Advertisers* create ads. They are the revenue sources of online advertising. During an ad delivery process, ad networks play the role of match-makers to bring together publishers and advertisers. Large ad networks often provide platforms (e.g., Google Display Network [2]) where advertisers can select publishers and specify targeted audience. Ad networks could also resell ad spaces in their inventory to other ad networks through ad syndication.

• *Audiences*, or users visit publisher pages and receive ad contents (e.g., ad banners). When they click these ads, they will be redirected to the corresponding advertiser Web sites.

In addition to these main actors, there are several other parties playing different roles in ad delivery. For example, *trackers* gather delivery statistics, which is important to the performance measurement of ad campaigns.

Ad delivery process. Figure 1 shows how these parties interact to deliver ads.

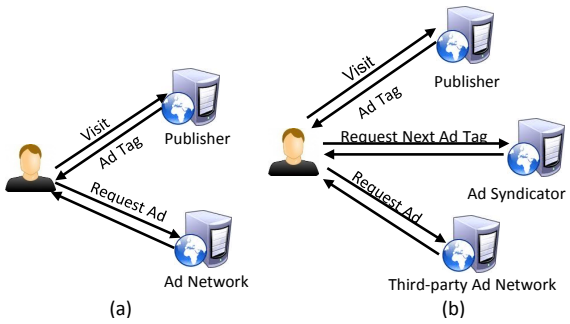


Figure 1: (a) Direct delivery (b) Ad syndication.

A publisher first embeds *ad tags* [14], which is a piece of HTML or JavaScript code, on its Web page for ad networks. Whenever a user visits the publisher page, the tags on the page will generate a request to an ad network for ad contents, including code, images, and others. The above dynamic process allows an ad network to customize the type of ads according to user geographic locations, behaviors, and activity histories. Alternatively, an ad network could also serve as an ad syndicator as shown in Figure 1(b), reselling ad spaces to other ad networks. When this happens, the code that the browser receives from the syndicator will fetch ad tags from third-party ad networks, which will either provide ad contents directly or further outsource the spaces to other parties.

2.2 How Malvertising Works: An Example

Online advertising has been extensively used by miscreants for malicious activities. To explicate how such malvertising works, we describe a real malicious ad campaign discovered by our study in June, 2011 and later confirmed by `BlueCoat Security Lab` in July, 2011 [17].

This is a fake Anti-Virus (AV) campaign that infected 65 publisher pages from June 21st to August 19th, 2011. One of them was the home page of `freeonlinegames.com`, an Alexa top 2404 Website. The page's ad tag first queried Google and DoubleClick, which referred the visitors to a third-party ad network `adsloader.com`. This ad network turned out to be malicious: it delivered an ad tag which automatically redirected the user's browser to a fake AV site and tried to trick the visitor to download a malware executable. Figure 2 illustrates this delivery process.

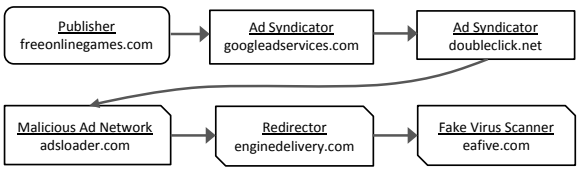


Figure 2: An example delivery chain of a fake AV campaign.

Figure 3: An ad delivered by adsloader.com.

What makes this campaign interesting is that its delivery path includes DoubleClick, a popular ad exchange network. The attackers set up a third-party ad network called `adsloader.com` (this domain name resembles `adloader.com`, held by a legitimate ad company) to syndicate with DoubleClick. When accessed by a victim, `adsloader.com` displayed an image (Figure 3).

Besides delivering an ad image, `adsloader.com` also injected a hidden iframe pointing to `enginedelivery.com`, which redirected users to `eafive.com` (a fake AV site), whose HTML code was classified by Forefront as TrojanDownloader:HTML/Renos.

After visiting the publisher with different configurations, we found that all of the involved malicious parties performed cloaking to evade detection. Specifically, `adsloader.com` never redirected the visitor from the same IP address to `enginedelivery.com` twice, and only did the redirection if the user agent was IE. It also checked a request's referrer field and did not inject the iframe when it was empty. The redirector `enginedelivery.com` did not send malicious contents to requests from certain IP ranges (e.g., Amazon EC2 IP ranges). Finally, the fake-AV Web site `eafive.com` attacked only IE-6 users. The attackers recruited in total over 24 ad networks, 16 redirectors, and 84 fake-AV scanners, and rotated them throughout the campaign. This strategy worked well: only 4 redirectors and 11 fake-AV sites were caught by Google Safe Browsing; none of the malicious ad networks were blocked.

This attack exhibits the following features:

• Each attack in this campaign requires three types of entities (ad-networks, redirectors, and fake-AV hosts) to work together.

• These entities could be controlled by different malicious parties. The Whois records [34] of the malicious ad networks are quite different from those of the redirectors and the fake-AV sites, suggesting that they may have been registered by different parties.

• All malicious domains were registered after 2010 and set to expire in one year, suggesting that they are registered by attackers within a short period.

These findings indicate that malvertising has distinctive infrastructure features. Such features, particularly those of the entities involved in an ad delivery process and their relations may provide valuable information for malvertising detection.

2.3 Attacks Leveraging Malvertising

We consider the following three categories of attacks in our research. All of them leverage the ad-delivery infrastructure to conduct malicious activities.

• *Drive-by download:* Such attacks exploit the vulnerabilities of browsers or plugins using dynamic contents in JavaScript or Flash.

• *Scam and phishing:* These attacks include fake-AVs or others that attempt to trick users into disclosing sensitive information, e.g., usernames, passwords, and bank account numbers.

• *Click-fraud:* Publishers routinely embed advertiser URLs with clickable links on their Web pages as *contexual ads*. Only when a user clicks such a link will the user be redirected to an advertiser page. However, we find that attackers set up malicious publisher sites and redirect user traffic (e.g., via hidden iframes) to advertiser pages automatically without user awareness, thus generating fraudulent clicks [13, 23].

In all of these attacks, attackers store malicious contents on either their own Web sites or compromised sites. To attract victims, traditionally, attackers promote these sites via blackhat SEO techniques [20, 16] or spam campaigns [30]. As online advertising reaches a large user population today, attackers have started exploiting ad networks, including DoubleClick and Zedo, to launch attacks in different ways. For example, drive-by downloads, scams, and phishing typically exploit malicious advertisers or ad networks to reach victims, whereas click frauds often go through malicious or compromised publishers.

3. DATASET AND TERMINOLOGY

Our research focuses on the ad infrastructure, which links multiple ad-related parties during an ad delivery process. By *infrastructure*, we broadly refer to the collective set of entities involved, their roles in Web advertising, and their interactions and relationships with each other. Our goal is to identify distinguishing infrastructure-related characteristics and to leverage them for developing detection techniques. To this end, we crawl popular Web pages, which we call *publisher pages*, to measure and analyze *ad-redirection chains*. In this section, we describe our data collection process and define the concepts to be used throughout this paper.

3.1 Dataset Collection

To collect ad-related traces, we build a crawler as a Firefox add-on. We configure its user-agent string to make it look like IE-6 and have it automatically clear cookies after visiting a Web page. We deploy the crawler using 12 Windows virtual machine (VM) instances on 12 different IP addresses from 3 subnets. These instances continuously crawl the home pages of Alexa's top 90,000 Web sites from Jun 21st to Sep 30th, 2011. Our crawler visits each of the pages once every three days. During each visit, a browser refreshes a page three times, in an attempt to obtain different ads. Since we primarily study display ads, the crawler just follows the automatic redirections triggered by the visit and does not click on any links, including the ad links embedded in the crawled pages. Our crawler could thus miss the cases when the malicious code is triggered only when an ad link is clicked.

For each visited page, we record all network requests, responses, browser events, and the code retrieved. Then, we reconstruct ad redirection chains by identifying the causal relations among the set of HTTP requests (URLs) originated from the page. Recall the ad delivery process illustrated in Figure 1: the publisher's Web page

first redirects the audience's browser to an ad network, which either returns an ad directly or performs a further redirection. The redirections are typically implemented through JavaScript, HTML code, or HTTP redirection (e.g., through status code 302 in response). To reconstruct redirection chains, we can connect two HTTP requests through a request's Referral field (the page downloaded by Request A generates Request B) or the Location field of a request's response (Request A's response redirects the browser to URL B). However, for the redirections caused by scripts, we are unable to use Referrer and Location to establish such a causal relation. Our solution is to extract the URLs from the script code and match them to those used by the HTTP requests observed after the execution of the script: once a script is found to contain the URL to which the browser produces a request, we have reasons to believe that the request may come from the script. This approach fails when the script actually concatenates several strings to build a redirection link and therefore does not contain a complete URL. We address this problem by simply identifying the domain names from each script code and assume that follow-up requests to these domains are produced by the corresponding script. In this way, we obtain 24,801,406 unique redirection chains and 21,944,174 unique URLs during the data collection. A similar approach has also been used by Google Safe Browsing [27]. We acknowledge that our current way to build the redirection chains may be less effective in the presence of Javascript obfuscation, but this problem can be addressed through analyzing the behavior of the code dynamically, which has been used for XSS detection [24].

3.2 Node, Path, and Domain-Path

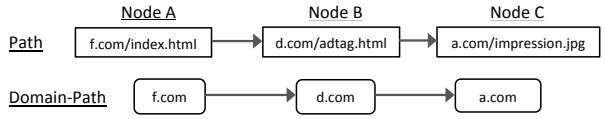


Figure 4: An example illustrating node, path, and domain path.

The large set of redirection chains provide us with a collective view on both the individual parties in advertising and the overall topologies of the entire infrastructure. Below we define the entities that we study in this paper.

- *Node:* We use the term *node* to refer to each URL encountered during the data crawling.
- *Path:* We call a reconstructed URL redirection chain a *path*. A path consists of a set of nodes (i.e., URLs), *ordered* by their redirection relations based on inferred causality.
- *Domain-path:* We observe that different crawls sometimes result in slightly different URLs along ad redirection (e.g., for user tracking purpose, or the delivery of different ads), but these URLs correspond to the same set of Web domains. So for each path, we extract its corresponding URL domains to build a unique *domain-path*. Note that one publisher may be associated with multiple domain paths.

The aforementioned concepts are illustrated in Figure 4. Publisher pages always correspond to source nodes. While paths describe the dynamic interactions between URLs, domain-paths are more stable and capture the business relationships between domains.

3.3 Role Marking

Not all the paths collected by our crawler are related to ads. To identify ad-delivery paths, we inspect individual nodes on each path using two well-known lists EasyList [26] and EasyPrivacy [26]. EasyList includes domains and URL patterns for ad-related hosts, and is used by the popular browser plugin Adblock plus [1] to block ads. EasyPrivacy is a list complementary to EasyList for identifying Web sites that track users. With these two lists, we further classify nodes as follows:

- *Publisher node:* We mark nodes from the publisher domains as *publisher nodes*. Publisher nodes are usually from the landing domains (the source nodes). However, they can appear at other locations on a path as well, for example, when they perform redirections. In our data, we find that 2.25% of the paths contain publisher nodes in the middle.
- *Ad node:* We label a non-publisher node as an ad node if it matches the features reported by EasyList or EasyPrivacy [26]. In addition, we label nodes showing images or SWFs [4] as ad nodes if they share a path with other identified ad nodes. These nodes were mostly used for delivering graphical ads.
- *Unknown node:* If a node is neither a publisher nor an ad node, we label it as unknown.

Paths	Nodes	Publisher Nodes	Ad Nodes	Domain-Paths
24,801,406	21,944,174	393,569	20,036,475	2,396,271

Table 1: Crawling statistics.

Accordingly, we treat a path as ad-related if it includes at least one ad node. Out of the 90,000 crawled publisher pages, 53,100 of them led to ad-related paths [2]. Among these paths, we marked 93.1% of the nodes as either publishers or ad nodes. Table 1 shows the statistics of the data collected and the ad-related roles marked.

3.4 Problem Statement and Challenges

Our goal is to broadly detect malicious and fraudulent activities that exploit display ads. In particular, if any node on an ad-delivery path performs malicious activities (e.g., delivering malicious content, illicitly redirecting user click traffic, etc.), we call the node a *malicious node*. Correspondingly, we call any path containing a malicious node a *malvertising path*, and the source node (i.e., the publisher's URL) of a malvertising path an *infected publisher*. Note that once we identify a malicious node, the following nodes on the same path are *not* always malicious. For example, when a malicious node cloaks, it may redirect a user to a legitimate Web site. In addition, click-fraud attacks use malicious nodes to redirect traffic to legitimate ad networks.

Malvertising detection is a challenging task. First, the partner relations of ad entities are often determined in real time by ad-exchange and are thus highly dynamic. From external observations, both legitimate and malicious ads can be delivered through multiple dynamic redirections, with new interactions coming up all the time, making it hard to distinguish malicious behaviors from legitimate ones. Further, this challenge cannot be effectively addressed by inspecting the contents of individual nodes or their features (e.g., URL or domain features): attackers not only use sophisticated code packing techniques to obfuscate content, but also compromise legitimate Web sites and turn them into malicious ad networks; it is thus difficult to differentiate between malicious and legitimate entities in isolation. Finally, malvertising attacks are of diverse categories (e.g., drive-by-downloads, phishing, and click frauds), each exhibiting different behaviors, making detection even harder.

To address these challenges, we perform a measurement study on the malvertising cases we encountered and compare them with legitimate cases. Based on our findings, we derive a simple and novel representation of the ad infrastructure that captures a variety of malvertising attacks in the wild. We present our measurement study and the detection methodology in the follow-up sections.

[2]Not all Alexa top Web site include ads on their home pages (e.g., http://www.google.com).

4. MEASUREMENT RESULTS

Using the dataset we collected, we analyze the malvertising activities and their infrastructure features in this section.

4.1 Malvertising Attacks Encountered

We scan all the nodes on the identified ad paths using the Google Safe-Browsing API and Microsoft Forefront 2010 to detect malvertising. If any node is flagged by either of the two scanners, we assume that it is a malicious node and flag its publisher as an infected publisher page. Among our data, Forefront detects 89 infected publisher pages and Safe Browsing detects 199. In total we identify 286 infected pages, with 543 malicious nodes coming from 263 domains, resulting in 938 malicious domain-paths.

We further classify attacks into three categories (drive-by-download, scam, and click fraud) as follows: if Forefront reports a node as "Exploit" or "Trojan", we label the attack as drive-by-download; if Forefront reports "Rogue", we treat it as scam. For the remaining cases, we manually examine the traces to determine the natures of the attacks.

Table 2 shows the statistics of identified malvertising attacks. We observe several distinguishing features. First, each of these three types of malvertising attacks takes a significant portion of all the attacks detected, suggesting attackers extensively exploit online advertising in multiple ways. Several publisher pages were associated with more than one type of attacks. For example, the porn Web site `privatepornclips.net` was exploited for both click frauds and drive-by-downloads. The domain-path via `gesttube.com` → `heatube.com` led to a pay-per-click ad network `clickpayz.com` for click fraud attacks, while domain-path `gesttube.com` → `sexyadultdating.net` led to drive-by-download attacks[3].

Second, the average malvertising path length is 8.11 nodes, much longer than the average crawled ad path length of 3.59 nodes, possibly due to both the existence of multiple entities (e.g., exploit servers and redirectors) and the use of ad syndication. We further investigate the correlations between malvertising and ad syndication in Section 4.3.

Third, the average life time of a particular malicious domain in our data is relatively short, ranging from 1 to 5 days, while the overall campaign can last for months (Section 2.2 shows an example campaign). Thus the individual malvertising domains can be more dynamic and harder to detect due to their transient nature and the use of domain rotations by attackers.

Finally, the infected publisher sites have large variations in their rankings at Alexa, suggesting that attackers target both large and small domains. Popular, trusted domains may also become victims. This feature is quite different from previously reported SEO attacks that primarily target small domains [16].

4.2 Properties of Malvertising Nodes

Through analyzing the malicious nodes captured by Safe Browsing and Forefront, we discover the following features that could be used to distinguish malicious nodes from legitimate ones.

Node roles: While a vast majority (93.1%) of the nodes on ad paths can be labeled as either a publisher or an ad node, most (91.6%) of the malicious nodes detected are marked as unknown. This comes with little surprise, as malicious nodes are often exploit servers whose URLs do not conform to well-known ad URL conventions.

Domain registration: The registration times of malicious node domains also differ significantly from the remaining ones. Figure 5 shows that most of the malicious domains expire within one year of registration. Further, many of them are newly registered in 2011.

[3]The URLs were flagged as "delivering malware" by the scanners. Our manual examination shows that they performed click frauds as well.

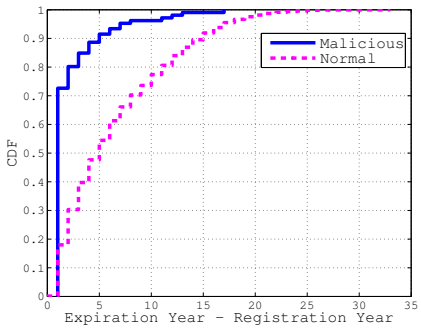
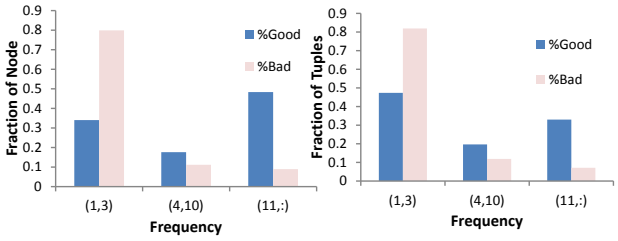


Figure 5: CDF of the durations between the registration dates and the expiration dates of Web domains.

Since malicious domains usually get blacklisted quickly, attackers may have no incentives to register long-living domains. In contrast, normal nodes have longer expiration dates as their business is expected to operate for years. This observation is more prominent for advertisting business: only 0.4% of legitimate ad nodes use newly registered domains comparing to 3.6% from legitimate none-ad nodes.

URL patterns: Many malicious domains belong to free domain providers such as `.co.cc`. Moreover, many of the exploit servers and redirectors have distinctive URL features. For example, the URL pattern `/showthread\.php\?t=\d{8}` matches the URLs of 34 different malicious nodes, suggesting that attackers have used templates or scripts to generate URLs.



(a) Node frequency (b) Pair frequency

Figure 6: Two frequency features.

In addition to the above features extracted from individual malicious nodes in isolation, we also observe the following two features that describe a node based on our global crawling results.

Node frequency: This metric measures the popularity and stability of node domains. For each node, we identify its domain and count the number of different publishers that are associated with this domain on each day. We then compute the total number of such occurrences over the days to find out the frequency of the node. Figure 6 (a) shows that most (nearly 80%) of the malicious nodes belong to the low frequency category, quite different from those within the legitimate category. This observation suggests that attackers usually create new ad networks or hijack small, unpopular ones, rather than directly targeting large, popular ad networks that are better managed and harder to compromise.

Node-pair frequency: This metric describes the stability of the business partnerships among different entities. We examine the frequency of two neighboring nodes on ad paths (referred to as *node pairs*) in a similar way by computing the corresponding domain pair popularity. Frequent pairs indicate stable partnerships (e.g., `youtube.com` to `doubleclick.net`). We find popular pairs are less likely associated with malicious nodes (Figure 6 (b)). In

	# of publisher pages	Avg path length	Avg malicious domain life time (days)	Max ranking	Min ranking
Drive-by-download	168	6.94	3.00	89814	314
Scam	66	6.52	1.21	85994	400
Click-fraud	63	12.61	5.75	89814	7659
All	286	8.11	2.96	89814	314

Table 2: Malvertising attacks captured by Google Safe Browsing and Microsoft Forefront from June to September.

contrast, malicious nodes are more likely to appear in new, infrequent pairs (e.g., `doubleclick.net` to `adsloader.com`).

The above two features are tightly associated with the ad infrastructure and the relations among different nodes. They are more robust to the attacker's possible counter strategies than individual node features. However, these features by themselves cannot be used straightforwardly for detection. For example, ad partnerships sometimes are determined in realtime by ad-exchange, so it is also common to see newly appeared, legitimate node pairs in Figure 6 (b).

4.3 Properties of Malvertising Paths

In addition to individual nodes, our measurement study further examines malvertising paths to understand the infrastructure behind those malicious activities.

	One ad network	Multiple ad networks
With DoubleClick	8	93
Without DoubleClick	330	507

Table 3: The number of domain-paths vs. the number of ad networks on the malvertising paths.

The use of ad syndication: We find that 64% of the malvertising domain-paths involve more than one ad networks on the paths (Table 3). These paths may be associated with ad syndication, where large ad networks such as DoubleClick resell ad spaces to small ad networks that are more vulnerable. Indeed, we find that 86 well known, legitimate ad networks, including `doubleclick.com`, `openx.com` and `admeld.com`, are tricked into referring malicious ad networks to Web clients. Out of the 101 malvertising domain-paths involving DoubleClick, there exist only 8 domain-paths where DoubleClick directly connects to a malicious node; the remaining ones all involve multiple ad networks and are likely caused by ad syndication.

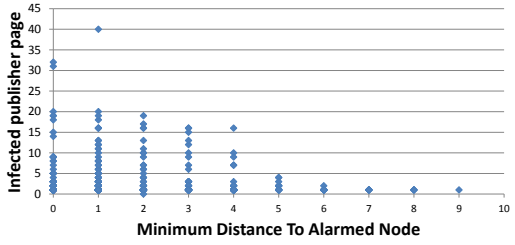


Figure 7: For each node on the malvertising paths, the minimum distance to a node detected by Safebrowsing or Forefront (alarm node) vs. the number of infected publishers that it is associated with.

Path distances among malicious nodes: Section 4.1 shows that malvertising paths are usually longer. This finding is consistent with previous observations [27]. However, we find that longer paths are not solely caused by ad syndication as reported before, since malvertising paths tend to include multiple nodes whose roles are unknown. These unknown nodes are often close in distance to the malicious nodes detected by Safe Browsing or Forefront, suggesting they are also suspicious. Specifically, we find 15.53% of the known malvertising paths include 3 consecutive nodes, all with unknown roles. In contrast, only 0.23% of the remaining paths have such cases. However, the exact positions of malicious nodes on these paths differ in different types of attacks (as shown Figure 14 in Appendix D).

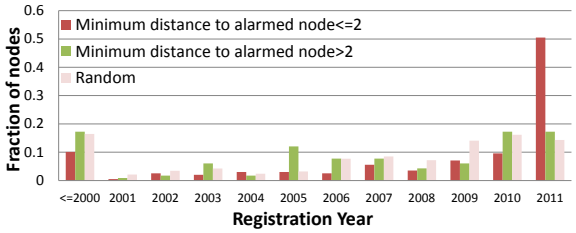


Figure 8: Node registration dates.

Figure 7 further shows that the closer a node stands to a malicious node, the more likely it is involved in multiple malvertising domain-paths. Since the detected malicious nodes are often redirectors or exploit servers, their neighboring nodes are also likely part of the malvertising infrastructure. We further inspect the registration dates of the neighboring nodes that are within 1 or 2 hops to the malicious nodes. Figure 8 shows that a large fraction of such nodes are newly registered in 2011. As a comparison, we also show in Figure 8 the registration date distributions of two other sets of nodes: the first includes the set of nodes that are at least 3 hopes away from a reported malicious node, and the second is a set of randomly sampled nodes. In both cases, fewer than 20% of them are newly registered.

4.4 Summary of Findings

Our measurement study shows that common node features, such as node roles and domain registration times, do help differentiate malicious ad nodes from legitimate ones to some extent. However, using these features in isolation is not reliable for detection. Even when they are used in combination on the individual nodes, the differentiation power is still limited (see Appendix A).

On the other hand, ad redirections also have unique conventions and characteristics that are different from typical Web site redirections. When we combine node features with ad paths, they become more distinctive for identifying attacks. For example, the roles played by different legitimate nodes (e.g., publishers, ad networks, and trackers) and their orders are not completely random. It is unusual to observe multiple consecutive nodes, completely unrelated with ads, staying together along the redirection chain of a normal ad. We also find that newly registered ad domains are much rarer than newly registered normal Web sites. So studying the topology and interactions among nodes, combined with their features, provides great opportunities for detection.

Finally, the observation that malicious nodes tend to stay together is helpful for detection. It suggests that we do not need to go beyond short path segments for detection—immediate neighbors often provide rich information for characterizing malvertising activities.

5. MALVERTISING DETECTION

In this section, we present the design and implementation of our system, called *MadTracer*, for detecting malvertising activities. Our measurement findings motivate us to explore ad-redirection paths, annotated with rich node features, to represent the underlying ad topology. Previous work has also studied and measured the characteristics of malicious Web redirection chains (e.g., [27]). The question is how to leverage the topologies and the interactions among ad nodes for detection.

Although we find malicious ad paths tend to be longer than normal ad paths, directly relying on the entire redirection paths for detection has two problems. First, a malicious path usually has mixed malicious and legitimate nodes. The presence of legitimate nodes adds noise to detection, especially when there exist multiple of them playing different roles, e.g., publishers, ad networks, and trackers. Second, the locations of the malicious nodes on a path are usually not fixed and we have encountered different cases in our study. For example, in drive-by-downloads, the malicious nodes often locate at the path tails. In click-fraud attacks, the malicious nodes usually locate in the middle of a path, between legitimate publishers and legitimate pay-per-click ad networks. Such diversified path behaviors add additional complexity in detection.

On the other hand, exploring simple, lightweight short ad-path segments holds great promise. Given malicious nodes usually stay close to each other on a path, as shown in Section 4.3. using short path segments mitigates the noises introduced by the presence of legitimate nodes. In addition, they are cleaner representations that eliminate the requirement of precisely identifying malicious node positions. Finally, such a formulation significantly reduces the complexity of our problem space and allows efficient solutions. While we do lose some information regarding the knowledge of entire paths, we find that short path segments are often sufficient to characterize the interactions among malicious entities. We show in Section 6 that such a representation works effectively in practice.

MadTracer consists of two major components. The first component identifies malvertising paths by analyzing ad paths and their features. The second is an analyzer component that intensively monitors the infected publisher pages, so as to study cloaking techniques and to expand our detection results. Figure 9 shows the architecture of MadTracer.

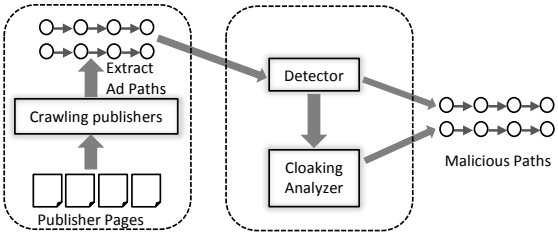


Figure 9: The infrastructure of MadTracer.

5.1 Detection Methodology

Our detection technique is based on analyzing annotated ad path segments. For each segment, we annotate every node with a set of attributes, including node popularity, the role in ad delivery, the domain registration information, and URL properties. These features, when applied to individual nodes, is not reliable for detection as we will show in Appendix A, but they add value to detection when they are combined with the topology information.

We adopt a statistical learning framework based on decision trees to automatically generate a set of detection rules. Figure 10 shows the process flow. Given input ad paths, MadTracer first annotates each node with a set of predefined attributes. It then extracts path

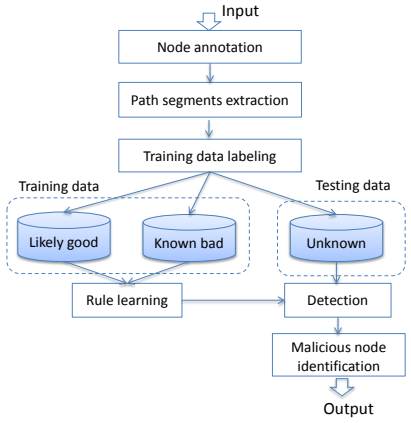


Figure 10: The process flow of malicious ad detection.

segments and selects a subset of them as training data to learn rules. When new data arrive, MadTracer apply the set of already learned rules. Meanwhile it also generates new rules periodically. We elaborate the details below.

Node annotation. Based upon our measurement study, we use the following four types of attributes to annotate a node:

Frequency attributes: The popularity of nodes and node pairs across the entire ad topology provides information about the scales of the corresponding Web sites and their business pairing relationships. MadTracer computes the frequency of every node and node pair in the collected data, and classifies them into the *popular* and *unpopular* categories according to an occurrence threshold (which was set to 10 in our research). For a pair of consecutive nodes A→B, we mark the pair's popularity attribute at B.

Role attributes: As discussed in Section 4, a node belonging to a known publisher or an ad-related entity is much less likely a malicious one. In contrast, those with unknown roles are more suspicious. Therefore, MadTracer annotates individual nodes with the roles they played using EasyList and EasyPrivacy, as described in Section 3.

Domain registration attributes: Our measurement suggests that domain registration and expiration dates can help differentiate legitimate domains from malicious ones. Therefore, for each node, MadTracer queries the Whois server [34] to obtain the *registered lifetime* of its domain, i.e., the duration between its registration and expiration dates. We label a domain's lifetime as *long* if it is longer than one year and *short* otherwise.

URL attributes: Section 4.2 shows that some malicious nodes can be characterized by the unique features of their domain names and URLs. We use the following two methods to derive such features. First, we identify free domain providers in our data (e.g., `co.cc`); many of them are also widely used by spammers [11]. MadTracer annotates all the nodes from such domains as *domain-suspicious* and others as *domain-normal*. Second, we derive URL regular expressions for each malvertising campaign captured in the training data. Similar to the previous approach using URL features to detect SEO campaign [16], we extract lexical features from URLs alarmed by Safebrowsing and Forefront, and cluster the URLs that share the same features. The lexical features include subdirectory name, filename, and argument name. Then we manually generate regular expressions from the URL clusters. Note that this step can be automated using regular expression generation tools such as AutoRE [35]. In total we generate 37 URL regular expressions. If a node matches any of the 37 regular expressions, MadTracer annotates it as *url-suspicious* and otherwise as *url-normal*.

Ad path segment extraction. After annotating nodes, MadTracer proceeds to derive ad path segments. Given our interest is in the ad-delivery topology rather than specific publishers, MadTracer first removes all the known publishers from the input paths. Furthermore, if a set of consecutive nodes from the same domain share identical attributes, MadTracer merges them into one node. After this preprocessing, MadTracer extracts all possible 3-node path segments from the input paths. For example, from a path $a \rightarrow b \rightarrow c \rightarrow d \rightarrow e$, we can generate 3 segments: $a \rightarrow b \rightarrow c$, $b \rightarrow c \rightarrow d$ and $c \rightarrow d \rightarrow e$. If a path is shorter than three hops, we use empty nodes (with all `null` attributes) as its prefix.

We find that 3-node path segments work well empirically. As discussed earlier, longer path segments might carry more information, but they tend to be too specific and often involve legitimate nodes. The classification complexity also grows substantially with longer segments, as the possible number of node attribute combinations will grow significantly.

Training data selection. MadTracer uses a "known bad" dataset and a "likely good" dataset to generate detection rules. The "known bad" set includes the malvertising paths detected by Safe Browsing or Forefront. The second dataset contains all the remaining paths that correspond to long-lasting domain-paths in our data. The rationale is that individual malvertising domain-paths are usually short-lived, with the average lifetime being a few days as shown in Section 4.1. Therefore, if a domain-path segment has a long life-span, it reflects legitimate, stable business partnerships. So MadTracer treats domain-paths whose lifetimes (between their first and last appearances) are longer than one month as "likely good". Although this approach does not guarantee that the training set does not include any malicious nodes, it significantly reduces the chance for such contamination to happen.

Learning and detection. MadTracer generates a set of detection rules via building a full decision tree. Since each node has 6 different attributes, the entire decision tree can have a large number of leaf nodes. We take advantage of the relatively small "known bad" dataset and prune the tree by selecting a subset of the leaf nodes that can detect at least one malicious node from the training data. We then sort them in an ascending order according to their false positive rates on the "likely good" training data, and return a set of l leaf nodes whose rules each result in a false positive rate no higher than a pre-defined threshold fp_α (set to 0.02% in our research). Finally, we merge these selected rules along the tree structure (e.g., if a certain attribute is agnostic, we remove it from the rules) to obtain a set of more compact detection rules.

Detection can take place either online during crawling, or offline periodically. In the detection phase, MadTracer does not require Safebrowsing or Forefront. It uses the already produced rules to match against each ad-path to be detected. If a path segment matches any of the learned rules, MadTracer reports the corresponding path as a malvertising path, and mark the corresponding publisher as infected. The detected publishers are then handed over to the analyzer component for further monitoring and analysis.

5.2 Attack Monitoring and Analysis

For each alarmed publisher page, the analyzer intensively crawls it with different configurations in order to conduct further analysis, including understanding cloaking and identifying more malicious nodes and paths.

We deploy 12 VMs at three different geo-locations to perform the monitoring [4]. These VMs monitor already detected publishers using different browser user-agent configurations (IE 6 and Firefox 3.6) and cookie clearing strategies ("always clear cookies" and "always store cookies"). Each VM continuously visits the entire set of detected publisher pages one by one, each time refreshing a page three times consecutively before moving on to the next one. As soon as it goes through the entire list, it restarts this process from the beginning. This monitoring allows the discovery of new malvertising domain-paths, which we report in our evaluation study. The analyzer also gathers data useful for understanding cloaking strategies. Finally, both the detected and the newly discovered malvertising paths can serve as new learning data to adjust detection rules. Although the scale of our current cloaking study is relatively small, a few interesting observations have already emerged (e.g. , the preferences on browser types). Appendix C reports our findings for this study.

[4]These VMs are deployed in Chicago, San Diego and Florida.

6. EVALUATION RESULTS

We evaluate MadTracer using four-month data. In this section, we first categorize the detected attacks and validate them. Then, we summarize newly identified malvertising characteristics and their cloaking strategies. Finally, we compare our detection results with those produced by existing methods.

6.1 Training and Detection Results

Dataset	# of 3-node path segments
Training-known-bad	1,254
Training-likely-good	9,346,436
Testing-likely-good	9,346,436
Testing-Jun-Sep	842,985
Testing-Oct	7,954,268

Table 4: Training and testing datasets

	# Total	#FP	%FP
pages	51,444	57	0.11%
domain-paths	1,198,136	899	0.075%

Table 5: False positive rates (Testing-likely-good dataset).

	# detected	#FP	%FD
scam pages	56	0	0.00%
drive-by-download pages	172	17	9.88%
click-fraud pages	155	17	10.97%
all pages	326	29	8.90%
scam domain-paths	104	0	0.00%
drive-by-download domain-paths	1171	73	6.23%
click-fraud domain-paths	4221	173	4.10%
all domain-paths	5496	246	4.48%

Table 6: Detection results (Testing-Jun-Sep dataset).

Our training data are derived from the traces collected between Jun 21st, 2001 and Sep 30th, 2011. The data are classified into "likely good", "known bad", and "unknown" categories using the method in Section 5.1. We further divide the "likely good" data into two equal-size subsets. One of them (*Training-likely-good*) and the "known bad" dataset are used for training. The other (*Testing-likely-good*) is for evaluating false-positives (FP). The "unknown" data serves as one testing dataset (*Testing-Jun-Sep*) for studying the coverage of MadTracer, together with another testing set (*Testing-Oct*) crawled from Oct 1st to Oct 30th, 2011. Table 4 summarizes these datasets.

MadTracer generates 82 rules from the training data. We first check the false-positives caused by these rules using the subsequences in *Testing-likely-good*, and measure the false positive rate. Here the false positive (FP) rate is defined as $N_{FP}/(N_{FP}+N_{TN})$, where N_{FP} denotes the number of false positives and N_{TN} is the number of true negatives. MadTracer detects 0.11% pages and

	#MadTracer	#S&F	#FP	#S&F-MadTracer	#MadTracer-S&F	FD(%)	New findings (%)
scam pages	12	0	0	0	12	0.00%	100.00%
drive-by-download pages	216	104	20	8	120	9.26%	51.85%
click-fraud pages	89	7	13	1	83	14.61%	92.13%
all pages	291	111	32	9	189	11.00%	61.86%
scam domain-paths	23	0	0	0	23	0.00%	100.00%
drive-by-download domain-paths	627	216	87	20	431	13.88%	65.55%
click-fraud domain-paths	3422	42	125	26	3406	3.65%	98.77%
all domain-paths	4072	258	212	46	3860	5.21%	93.66%

Table 7: Detection results (Testing-Oct dataset). "MadTracer" denotes our detection results. "S&F" denotes the results detected by Safe Browsing and Forefront. The "New findings" column computes the percentage of attacks detected by MadTracer over the total number of attacks detected by MadTracer, SafeBrowsing, or ForeFront.

0.075% domain-paths in the set, which are supposed to be false alarms. This indicates that the FP rate introduced by our approach is very low. The details of the study are shown in Table 5.

We then evaluate the performance of MadTracer on *Testing-Jun-Sep* and *Testing-Oct*. MadTracer detects 617 infected publishers and 9,568 unique malve-rtising domain-paths in total with a false detection (FD) rate around 5%. We define the FD rate here as the number of falsely detected domain-paths or pages over the total number of detected domain-paths or pages: that is, $N_{FP}/(N_{FP} + N_{TP})$, where N_{TP} is the number of true positives. Given 53,100 out of 90,000 crawled publisher Web pages have display-ad-related paths, we observe from our data that over 1% of the top Alexa home pages lead to malvertising. Since these are well reputable domains with a high volume of traffic, malvertising through them could have reached a large victim user population. Tables 6 and 7 elaborate the results.

6.2 Attack Classification and Validation

MadTracer is designed to capture the common features of malvertising. It does not distinguish the type of attacks (scam, drive-by downloads and click frauds) for the suspicious paths it detects. To validate its detection results, we first classify those detected cases heuristically and then work on the cases in individual categories according to the suspicious behavior that they exhibit. This validation process is elaborated below.

Scam. For malicious paths that trigger scam popup windows, we place them in the *likely scam* category, as popup windows are frequently related to scam attempts. Those images typically display catchy contents such as "Your computer is infected" or "You are the winner". Besides fake-AV, we also find another type of scam—lottery phishing, as shown in Table 8. Lottery phishing attacks redirect a user's browser to a phishing page, which announces that the visitor has won a big prize (e.g., Figure 11). Then the user is asked to fill in private information such as her cell phone number and bank account numbers. The information collected can be sold to a third party or used for identity theft.
Validation: We manually go through the images in the popup windows to validate these scam cases, as their number is small.

Drive-by-downloads. For malicious paths that do not trigger popup windows, we analyze the locations of the detected 3-node segments. If such a path segment appears after ad nodes (identified by EasyList and EasyPrivacy) on the path, it corresponds to the situation where attackers redirect users from ad networks to malicious servers, so we classify it as a likely drive-by-download.
Validation: To validate these attacks, we first scan all the nodes involved using Safe Browsing and Forefront. For the remaining ones, we submit them to Microsoft Forefront for in-depth analysis. They confirmed that a vast majority of the detected path segments contain malicious executables using new signatures. We conservatively treat all unconfirmed cases as false positives. For the ones detected by Forefront, we notice that more than half of them are under the category Exploit:JS/Blacole. This type of exploit is generated by the Blackhole exploit kit, which is widely used by attackers to set up exploit servers [25]. This toolkit also includes malicious code exploiting a number of recent vulnerabilities in Java and Adobe PDF.

Click-fraud. We find that the remaining cases are mostly related to click fraud. In contrast to legitimate publishers who display ad links (pointing to advertiser's landing pages) that users can click, fraudulent or compromised publishers redirect user traffic through pay-per-click (PPC) ad networks to ad landing pages automatically, without showing the ads to users and without the need of user clicks. Up to our knowledge, this type of click frauds has not been reported before. Safe Browsing and Forefront fail to detect most of such click fraud attacks since these attacks do not involve malicious executables. We present the details of the attacks in Appendix B.
Validation: To validate such attacks, we examine the detected ad paths based on two prominent properties of click fraud. First, we examine whether a publisher page contains an invisible iframe [13] to redirect user traffic automatically without the need of user clicks. Second, we check whether the path eventually reaches an ad landing page through a PPC ad network. If a path has both properties, it means that the publisher page successfully redirects a browser to an ad landing page without actual user clicks, which fulfills a fraudulent click. However, not all click frauds are successful, some of them may be detected by the PPC networks, so the traffic could not reach the final ad landing pages. To validate such failed cases, we compare their paths with the successful click-fraud paths. If they went through the same redirection domain chains as the successful ones, we regard them as likely click frauds as well.

Tables 6 and 7 list the detailed evaluation results based on the above validation process. The overall FD rate of our detected malvertising domain-paths is 4.48% for the Testing-Jun-Sep dataset and 5.21% for the Testing-Oct dataset. We present the details of our findings and the study on cloaking techniques in Appendix C.

	# of publisher pages	# of domain-paths
Lottery	16	63
Fake AV	52	64

Table 8: Detected phishing attack break-down

6.3 Comparison with Existing Techniques

We compare our detection results with those obtained by using URL and domain attributes only. We find 10.2% of the detected malicious domain-paths display suspicious URL patterns. Thus compared to URL-based approaches, MadTracer can significantly increase the detection coverage.

Compared with Safe Browsing or Forefront, our method does miss 46 domain-paths detected by them. However, for the attacks that were successfully detected by MadTracer, our approach catches them earlier than existing solutions. Specifically, throughout Octo-

Figure 11: The lottery scam page.

ber, we ran MadTracer Safe Browsing, and Forefront on the traces collected from the beginning of the month on a daily basis. We find that Forefront usually detects malicious domains on the same day as our approach, but Safe Browsing on average needs 10.5 more days before it reports the domain-paths that we caught. Figure 12 shows a histogram that illustrates Safe Browsing's delays in detection. We find that several malvertising domain-paths in our Testing-Jun-Sep dataset detected by our approach were not reported by Safe Browsing until October, which introduces significant delay in taking measures to stop these ongoing attacks.

The early detection ability and the higher coverage of our approach demonstrate the power of detection using ad paths and rich node attributes. By focusing on the malvertising infrastructure instead of malicious ad contents, MadTracer has the ability to detect new, stealthy malvertising activities that slip under the existing malware scanners.

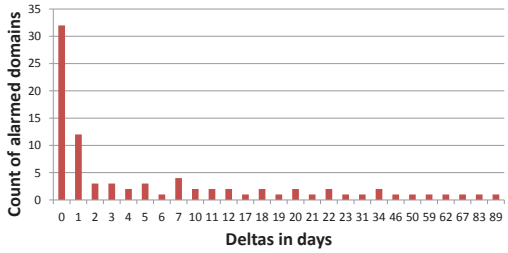


Figure 12: Early detection results.

7. RELATED WORK

Research on malvertising. Malvertising is an emerging threat but grows fast in recent years [9]. Prior research on this threat mainly focuses on controlling the behavior of ads in order to prevent malvertising (e.g., [19]). However, these approaches usually cannot defend against common attacks such as drive-by-download, and they also requires publishers to change their Web sites.

More general static and dynamic analysis techniques (e.g., [7] and [21]) could be applied to detect drive-by-download. An ad network could restrict and sanitize dynamic contents using static verifiers such as ADSafe [8] and its improvements (e.g., [10, 19, 12]). These countermeasures raise the bars for attackers who directly upload malicious contents to legitimate ad networks. But they could be easily circumvented by either sophisticated packing and anti-emulation techniques, or the use of malicious ad networks through ad syndication.

Ad syndication allows attackers to directly inject malicious code into a browser without being examined. Previous study [27] showed that ad syndication is a popular way to distribute drive-by-downloads. In our study, we move one step further to understand the detailed properties of malvertising paths, including the roles of each entity along the paths and the relationships among them. Our work complements the existing defense mechanisms. It also allows us to detect a broader set of other malicious advertising behaviors such as phishing and click frauds in a lightweight fashion.

Stone-Gross et al. [28] recently reported a study on fraudulent activities in online ad exchange based on traffic collected from an ad network. Different from our work, they have not investigated the topology of malvertising. Wang et al. [33] studied ad distribution networks and their properties. Their focus is on network performance and user latency, while we focus on the implications of ad network topologies for attack detection.

Our detection approach is based on analyzing 3-node ad path segments. Previous work has leveraged the n-gram model for predicting the next item in a sequence [5] or clustering malware samples [6]. Instead of exploiting the n-gram similarly as previous work, we work with annotated n-grams with rich node attributes. We reformulate the malvertising detection problem ad network topologies. From this perspective, we contribute by proposing a new presentation of topology using simple n-grams as well as demonstrating its effectiveness.

Research on other attack channels. In addition to online advertising, Blackhat SEO campaigns and spam emails are two other popular methods for attracting naive Web users. Recent work has studied the properties of these attacks and proposed a few detection strategies [16, 20, 15, 18]. Compared to SEO and spam, malvertising has received relatively less attention so far, yet it may pose a much more serious threat to Web security for two reasons. First, attackers may infiltrate large ad networks and thus infect top ranking Web sites with more visitors. Second, attackers could specify audience profiles at their choice through advertising agreements, and target attacks at the most vulnerable populations (e.g., grandparent visitors). Previous work has also shown the effectiveness of leveraging URL features in detecting redirectors [37] and compromised servers [16]. In our case, we find that using URL features alone is not sufficient, though it does provide a useful signal that can augment the topological information for detection.

8. DISCUSSION

Our study shows that malvertising is a severe problem on the Internet. By crawling just the top 90,000 Alexa home pages (among them 53,100 are publisher pages), we find that more than 1% of these well-maintained sites have been exploited to deliver malicious contents or to conduct fraudulent clicks. Considering our crawling scale is small, the actual malvertising problem can be more severe. This study calls for the research community to pay more attention to the malvertising problem.

Towards detection, we make a first step toward examining topologies and develop a method based on analyzing 3-node path segments. We demonstrate initial success in this direction with real data and a wide set of real attacks detected. On the other hand, we have not incorporated other useful features into our design, path length in particular, not to mention the whole topology of ad networks as a graph that could be used to achieve more effective detection. Further study on these issues is an interesting direction for future research.

The evaluation results show that MadTracer can detect a large number of malicious advertising cases, with an FD rate round 5%. We aim to detect as many malvertising cases as possible, instead of sacrificing the true-positive rate for a low FD rate. For end users, blocking malicious ads is perhaps more important than mistakenly blocking legitimate ads. This is different from detecting other ma-

licious activities such as spam, where flagging legitimate emails as spam bears more serious consequences.

To evade detection, attackers may exploit the node features that we adopt, e.g., by modifying URL patterns or using compromised old domains instead of registering new domains. Those attempts, however, should be less effective against MadTrace than approaches that just look at individual nodes. By exploring the ad infrastructure, MadTrace forces the attackers to change a sequence of nodes and their relations, which can be a hard task as those nodes may be controlled by different malicious parties within the underground ecosystem [31]. Also, faking ad-specific features that we utilize can be more difficult than it appears to be. Take the role feature as an example: the attacker who assigns an ad-related URL to a compromised non-ad host could risk exposing that host, due to the discrepancy between what the host was and what it looks like now. On the other hand, further research is needed to better prepare our approach for these evasion attacks.

We envision that MadTracer can benefit both service providers and Web users in multiple aspects. Large ad networks can use MadTracer to identify fraudulent activities, compromised and malicious syndicators, and infected publishers. The detected malicious ad contents can be fed into anti-virus systems to generate new content-based attack signatures. Finally, a browser-based protection mechanism can utilize the knowledge of malicious ad paths and their topological features to raise an alarm when a user's browser starts to walk down a suspicious ad path, protecting the user before she reaches an exploit server.

9. CONCLUSION

Today's Web advertising is permeated by malicious ads, which pose a serious threat to the Web users and legitimate businesses. This paper reports our measurement study for better understanding the infrastructure for delivering malicious ads. Based on a large-scale Web crawling, we reveal the gravity of the threat. We show that such attacks infected hundreds of publisher pages and infiltrated major ad networks including DoubleClick. The insights gained through the measurement study leads us to develop a new topology-based detection system—MadTracer. Our evaluation shows that MadTracer works effectively against real-world malvertising activities: it caught 15 times as many malicious domain-paths as Google Safe Browsing and Microsoft Forefront combined, and also discovered several large-scale malvertising campaigns, including a new type of click-fraud attack. A more detailed summary of our findings will be released on `www.madtracer.org`. Our work demonstrates that topology-based detection holds a great promise to more effectively mitigate malvertising threats.

Acknowledgements

We thank anonymous reviewers for their insightful comments. We are grateful for the help provided from Microsoft Forefront for attack analysis. IU authors also acknowledge NSF CNS-1017782 and CNS-1223477 for the support.

10. REFERENCES

[1] Adblock plus. `http://adblockplus.org/en/`.

[2] Display network google ads. `http://www.google.com/ads/displaynetwork/`.

[3] Wordpress, blog tool, publishing platform, and cms. `http://wordpress.org/`.

[4] Adobe. Adobe flash platform. `http://www.adobe.com/flashplatform`, 2011.

[5] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479, 1992.

[6] S. K. Cha, I. Moraru, J. Jang, J. Truelove, D. Brumley, and D. G. Andersen. SplitScreen: enabling efficient, distributed malware detection. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, NSDI'10, page 25, Berkeley, CA, USA, 2010. USENIX Association.

[7] M. Cova, C. Kruegel, and G. Vigna. Detection and analysis of drive-by-download attacks and malicious javascript code. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 281–290, New York, NY, USA, 2010. ACM.

[8] D. Crockford. Adsafe. `http://www.adsafe.org`.

[9] B. Edelman. Benjamin edelman - publications. `http://www.benedelman.org/publications/`, July 2012.

[10] M. Finifter, J. Weinberger, and A. Barth. Preventing capability leaks in secure javascript subsets. In *NDSS*. The Internet Society, 2010.

[11] D. Fisher. Google removes .co.cc subdomains over phishing, spam concerns. `http://threatpost.com/en_us/blogs/google-removes-cocc-subdomains-over-phishing-spam-concerns-070611`, 2011.

[12] S. Ford, M. Cova, C. Kruegel, and G. Vigna. Analyzing and detecting malicious flash advertisements. *Computer Security Applications Conference, Annual*, 0:363–372, 2009.

[13] M. Gandhi, M. Jakobsson, and J. Ratkiewicz. Badvertisements: Stealthy click-fraud with unwitting accessories. *Journal of Digital Forensics Practice*, 1(2), 2006.

[14] Google. What is an ad tag? - doubleclick for publishers help. `http://support.google.com/dfp_premium/bin/answer.py?hl=en&answer=1131465`.

[15] S. Hao, N. A. Syed, N. Feamster, A. G. Gray, and S. Krasser. Detecting spammers with snare: spatio-temporal network-level automatic reputation engine. In *Proceedings of the 18th conference on USENIX security symposium*, SSYM'09, pages 101–118, Berkeley, CA, USA, 2009. USENIX Association.

[16] J. P. John, F. Yu, Y. Xie, A. Krishnamurthy, and M. Abadi. deseo: combating search-result poisoning. In *Proceedings of the 20th USENIX conference on Security*, SEC'11, pages 20–20, Berkeley, CA, USA, 2011. USENIX Association.

[17] C. Larsen. Busting a big malvertising / fake-av attack. `http://www.bluecoat.com/security/security-archive/2011-07-25/busting-big-malvertising-fake-av-attack-0`, July 2011.

[18] K. Levchenko, N. Chachra, B. Enright, M. Felegyhazi, C. Grier, T. Halvorson, C. Kanich, C. Kreibich, H. Liu, D. McCoy, A. Pitsillidis, N. Weaver, V. Paxson, G. M. Voelker, and S. Savage. Click Trajectories: End-to-End Analysis of the Spam Value Chain. In *Proceedings of 32nd annual Symposium on Security and Privacy*. IEEE, May 2011.

[19] M. T. Louw, K. T. Ganesh, and V. N. Venkatakrishnan. Adjail: practical enforcement of confidentiality and integrity policies on web advertisements. In *Proceedings of the 19th USENIX conference on Security*, USENIX Security'10, pages 24–24, Berkeley, CA, USA, 2010. USENIX Association.

[20] L. Lu, R. Perdisci, and W. Lee. Surf: detecting and measuring search poisoning. In *Proceedings of the 18th ACM conference on Computer and communications security*, CCS '11, pages 467–476, New York, NY, USA, 2011. ACM.

[21] L. Lu, V. Yegneswaran, P. Porras, and W. Lee. Blade: an

attack-agnostic approach for preventing drive-by malware infections. In *Proceedings of the 17th ACM conference on Computer and communications security*, CCS '10, pages 440–450, New York, NY, USA, 2010. ACM.

[22] McAfee. Mcafee web gateway. `http://www.mcafee.com/us/products/web-gateway.aspx#vtab-Benefits`, 2011.

[23] B. Miller, P. Pearce, C. Grier, C. Kreibich, and V. Paxson. What's clicking what? techniques and innovations of today's clickbots. In *Proceedings of the 8th international conference on Detection of intrusions and malware, and vulnerability assessment*, DIMVA'11, pages 164–183, Berlin, Heidelberg, 2011. Springer-Verlag.

[24] F. Nentwich, N. Jovanovic, E. Kirda, C. Kruegel, and G. Vigna. Cross-site scripting prevention with dynamic data tainting and static analysis. In *In Proceeding of the Network and Distributed System Security Symposium (NDSS'07)*, 2007.

[25] A. NS. Blackhole exploit kit 1.0.2. `http://www.airdemon.net/blackhole.html`, 2011.

[26] R. Petnel. The official easylist web site. `http://easylist.adblockplus.org/en/`.

[27] N. Provos, P. Mavrommatis, M. A. Rajab, and F. Monrose. All your iframes point to us. In *Proceedings of the 17th conference on Security symposium*, pages 1–15, Berkeley, CA, USA, 2008. USENIX Association.

[28] B. Stone-Gross, R. Stevens, R. Kemmerer, C. Kruegel, G. Vigna, and A. Zarras. Understanding fraudulent activities in online ad exchanges. In *Proceedings of Internet Measurement Conference*, IMC '11, 2011.

[29] Sucuri. Mass infection of wordpress sites due to timthumb. `http://blog.sucuri.net/2011/08/mass-infection-of-wordpress-sites-counter-wordpress-com.html`, 2011.

[30] K. Thomas, C. Grier, J. Ma, V. Paxson, and D. Song. Design and evaluation of a real-time url spam filtering service. In *Proceedings of the 2011 IEEE Symposium on Security and Privacy*, SP '11, pages 447–462, Washington, DC, USA, 2011. IEEE Computer Society.

[31] TrendLabs. Follow the money trail. `http://blog.trendmicro.com/follow-the-money-trail/`, March 2012.

[32] A. VANCE. Times web ads show security breach. `http://www.nytimes.com/2009/09/15/technology/internet/15adco.html`, 2009.

[33] Y. Wang, D. Burgener, A. Kuzmanovic, and M.-F. Gabriel. Understanding the network and user-targeting properties of web advertising networks. In *ICDCS*, pages 613–622, 2011.

[34] Whois.net. Whois lookup - domain names search, registration, & availability. `http://www.whois.net/`, 2011.

[35] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov. Spamming botnets: signatures and characteristics. In *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, SIGCOMM '08, pages 171–182, New York, NY, USA, 2008. ACM.

[36] ZenithOptimedia. Global ad expenditure to return to pre-recession peak level this year. `http://www.zenithoptimedia.com/files/media/image/news/Press%20Release%20files/2011/July/Adspend%20forecasts%20July%202011.pdf`, 2011.

[37] J. Zhang, C. Seifert, J. W. Stokes, and W. Lee. Arrow: Generating signatures to detect drive-by downloads. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 187–196, New York, NY, USA, 2011. ACM.

APPENDIX

A. COMPARISON WITH INDIVIDUAL NODE CLASSIFIER

As a comparison, we evaluate the effectiveness of malvertising detection by applying the combination of features (as described in Section 5.1) on individual nodes for detection. We also use the same method and datasets as described in Section 5.1 and 6.1 for learning a set of detection rules.

Such individual node based classifier detects 20,533 domain-paths in the Testing-Jun-Sep dataset. However 17,614 of them are actually false-positives. Using the Testing-Oct dataset, the classifier detects 25,308 domain-paths with 23,140 of them being false-positives. For both datasets, the false detection (FD) rates are over 85%, and are significantly higher than those of MadTracer.

We sample a subset of the false positive domain-paths and find that most of them are detected because they either involve newly registered ad networks or ad networks that do not follow the URL patterns defined by EasyList. However, such ad networks all have legitimate portal sites and are unlikely to be hosted by attackers.

Meanwhile, the number of truly malicious pages and domain-paths that are successfully detected by the single-node based classifier is smaller than that by using MadTracer. We find that the rules that can detect malicious pages or domain-paths also incur a high false positive rate on the training data. So these rules are not selected by the learning framework for detection.

B. A LARGE CLICK-FRAUD ATTACK DETECTED

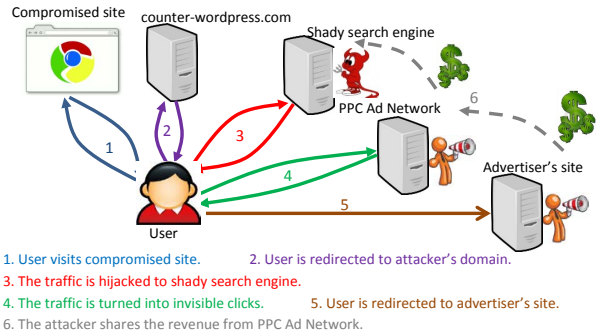


Figure 13: The flow graph of a click-fraud case.

Figure 13 shows the traffic flow of a big click-fraud campaign that we detected. The major entities involved in this campaign include compromised Web sites, attacker created shady (i.e.,fraudulent) search engines, legitimate pay-per-click (PPC) ad networks, and legitimate advertisers. Below, we present how this click-fraud attack exploits online advertising channels.

In this example, attackers control a large number of Web sites that are set up using old versions of WordPress [3] with known vulnerabilities. These sites were compromised [29] to redirect traffic to the attackers' domains (e.g., `counter-wordpress.com`). When a user visits any of these compromised Web sites, his traffic will be further redirected into multiple attacker-created *shady*

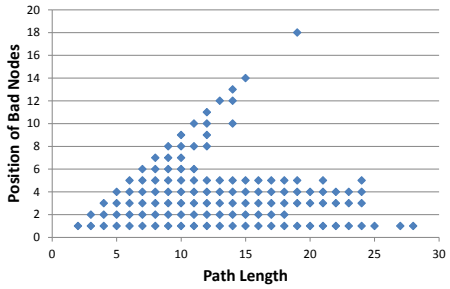


Figure 14: For the detected malicious nodes on the malvertising paths, their positions on the paths vs the corresponding path lengths.

search engines, which are actually a set of fraudulent domains (e.g., `getnewsearcher.com`) resembling search engines. The purpose of the shady search engines is to affiliate with legitimate PPC networks and to refer click traffic to them. Specifically, once user traffic reaches the shady search engines (without user awareness), it will be converted into fraudulent ad clicks and further redirect to the affiliated legitimate PPC networks through a set of redirectors, and eventually to advertisers.

After receiving the fraudulent click traffic, the advertiser pays the PPC network, which in turn pays the attacker-controlled shady search engines. In order to maximize revenue, attackers aggressively turn one user visit into multiple fraud clicks. In an extreme case, we observed that a user visit were turned into 37 clicks to 4 different PPC ad networks simultaneously. All the traffic redirection activities happen without user clicks or awareness, yet they significantly slow down the browser performance and negatively impact the user experience.

Using our approach, we identify 219 such shady search engines and 50 affiliated redirectors associated with this type of click frauds. Most of these cases were not detected by Safe Browsing or ForeFront as they are not used for delivering malicious contents. To evade detection by PPC ad networks, attackers intentionally redirects traffic through different shady search engines and redirectors so that the redirection paths look diversified and more legitimate. However, by examining the interactions among different entities along ad-related paths, our approach can successfully detect the hidden malicious infrastructure, even for these stealthy attacks.

C. FINDINGS AND CLOAKING STUDY

After validation, we revisit Tables 6 and 7 and notice several interesting observations. First, on average, each infected publisher page corresponds to multiple (15.5) malvertising domain-paths, where attackers rotate domains to evade detection. This attacker strategy, used for attack evasion, can actually help us discover more malicious nodes on the malvertising infrastructure by continued monitoring of infected publisher pages.

Compared with drive-by-download, click frauds are more dynamic. They infected a smaller number of publisher pages (138 in Jun-Sep and 76 in Oct) compared with drive-by-download (155 in Jun-Sep and 196 in Oct), the number of different domain-paths used for click fraud is significantly larger. Our manual investigation shows that attackers use a larger set of domains to serve as different roles for rotation. Though these domains usually do not exhibit distinguishing URL or domain features, they are detected by our approach because they usually form uncommon combinations in topology.

The detected suspicious cases were further fed to the monitoring component of MadTracer for continuous crawling. Our monitoring started in Oct 2011 and we reported our findings using the 126 detected publisher pages while the attack was still alive. We observe that all of the infected Web pages led to new malvertising domain-paths, with a coverage increase of 96.3%. In addition, we find that attackers often have strong preferences on browser settings. Internet Explorer (IE) is the most targeted browser type. Among the 126 pages, 95 of them deliver attacks successfully to IE, and 57 do not display malicious contents when visited by Firefox. The location preference, however, is not obvious from the monitoring results generated from different IP ranges, perhaps due to the fact that all of our VMs are located in the U.S.

D. THE POSITION OF MALICIOUS NODES ON MALVERTISING PATHS

Figure 14 shows the scatter plot in terms of the malvertising path lengths vs. the position of the alarmed nodes on the paths. Each point corresponds to one or more known malicious nodes in our measurement (multiple points may overlap at one position). The X-axis shows the path lengths, and the Y-axis shows the positions of the malicious nodes on the paths. We observe many points along the $Y = X$ line, meaning these malicious nodes are the last hop on the redirection chains. Such cases usually correspond to drive-by-download attacks, where the malicious nodes are the exploit servers. However, we also observe many malicious nodes locating in the middle of their redirection chains. Such cases are likely click-fraud attacks, where the malicious nodes serve as the purpose to redirect traffic from (legitimate or malicious) publishers to legitimate pay-per-click ad networks. These findings indicate that the positions of the malicious nodes on the ad paths are not fixed due to the diversified attack categories.