Detecting and Defending Against Fraud in the Underground Economy

Apostolis Zarras

Thesis submitted in partial fulfillment of the requirements for the

Masters' of Science degree in Computer Science

University of Crete
School of Sciences and Engineering
Computer Science Department
Knossou Av., P.O. Box 2208, Heraklion, GR-71409, Greece

Thesis Advisor: Prof. *Evangelos Markatos*

University of Crete
Computer Science Department

Detecting and Defending Against Fraud in the Underground Economy

Thesis submitted by
Apostolis Zarras
in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science

THESIS APPROVAL

Author: _____
Apostolis Zarras

Committee approvals: _____
Evangelos Markatos
Professor, Thesis Supervisor

_____
Dimitris Nikolopoulos
Associate Professor

_____
Sotiris Ioannidis
Principal Researcher, FORTH – ICS

Departmental approval: _____
Angelos Bilas
Associate Professor, Director of Graduate Studies

Heraklion, May 2011

Abstract

Online advertisements (ads) provide a powerful mechanism for advertisers to effectively target web users. These ads can be customized based on a user's browsing behavior, geographic location, and personal interests. There is currently a multi-billion dollar market for online advertising, which generates the primary revenue for some of the most popular web sites on the Internet. In order to meet the immense market demand, and to manage the complex relationships between advertisers and publishers (i.e., the web site hosting an ad), marketplaces known as ad exchanges are employed. These exchanges allow publishers (sellers of ad space) and advertisers (buyers of this ad space) to dynamically broker traffic through ad networks to efficiently maximize profits for all parties. Unfortunately, the complexities of these systems invite a considerable amount of abuse from cybercriminals, who profit at the expense of the advertisers.

In this thesis, we present a detailed view of how one of the largest ad exchanges operates and the security issues that it encounters from the vantage point of a member ad network. More specifically, we analyzed a dataset containing transactions for ingress and egress ad traffic from this ad network. In addition, we examined information collected from a command-and-control server used to operate a botnet that is leveraged to perpetrate ad fraud against the same ad exchange.

Περίληψη

Οι διαδικτυακές διαφημίσεις παρέχουν έναν ισχυρό μηχανισμό στους διαφημιστές ώστε να μπορούν να απευθυνθούν αποτελεσματικά στους χρήστες του Διαδικτύου. Οι διαφημίσεις αυτές, μπορούν να προσαρμοστούν με βάση τη συμπεριφορά περιήγησης του χρήστη, τη γεωγραφική του θέση, και τα προσωπικά του συμφέροντα. Υπάρχει σήμερα μια αγορά πολλών δισεκατομμυρίων δολαρίων, η οποία στηρίζεται στις διαδικτυακές διαφημίσεις και η οποία αποτελεί την κύρια πηγή εσόδων για μερικές από τις πιο δημοφιλείς ιστοσελίδες του Διαδικτύου. Προκειμένου να υπάρξει ανταπόκριση στις τεράστιες απαιτήσεις της αγοράς, καθώς και στη διαχείριση των σύνθετων σχέσεων μεταξύ των διαφημιστών και των εκδοτών (π.χ., ιστοσελίδες που φιλοξενούν μια διαφήμιση), προσλαμβάνονται "αγορές" γνωστές και ως "συναλλαγές διαφημίσεων". Οι "συναλλαγές διαφημίσεων" επιτρέπουν στους εκδότες (πωλητές διαφημιστικού χώρου) και στους διαφημιζόμενους (αγοραστές του εν λόγω διαφημιστικού χώρου) να δημοπρατήσουν δυναμικά κίνηση μέσω των δικτύων διαφήμισης, με σκοπό τη μεγιστοποίηση των κερδών από όλους τους συμμετέχοντες. Δυστυχώς, η πολυπλοκότητα των συστημάτων αυτών οδηγεί σε κατάχρηση του συστήματος από τους εγκληματίες του κυβερνοχώρου, οι οποίοι κερδοσκοπούν σε βάρος των διαφημιστών.

Στην παρούσα εργασία, παρουσιάζουμε μια λεπτομερή εικόνα για το πώς μια από τις μεγαλύτερες "συναλλαγές διαφημίσεων" λειτουργεί, καθώς επίσης και τα ζητήματα ασφάλειας που αντιμετωπίζει, παρατηρώντας τα από τη σκοπιά ενός μέλους του δικτύου διαφήμισης. Πιο συγκεκριμένα, αναλύουμε ένα σύνολο δεδομένων που περιέχει πληροφορίες για τις συναλλαγές που πραγματοποιούνται από το συγκεκριμένο δίκτυο διαφήμισης. Επιπλέον, εξετάζουμε τις πληροφορίες που συλλέγονται από ένα διακομιστή διοίκησης και ελέγχου που χρησιμοποιείται για τη λειτουργία ενός botnet, εξειδικευμένου στο να διαπράττει διαφημιστικές απάτες κατά της "συναλλαγής διαφημίσεων" που εξετάζουμε.

Acknowledgements

First of all, I would like to thank my supervisor, Prof. Evangelos Markatos, for his valuable guidelines in my academic steps in the field of Computer Science. I also feel grateful to Dr. Sotiris Ioannidis, for his invaluable help and cooperation over the last three years, and for a real commitment to my technical and professional growth.

My best thanks to all former and current members of the Distributed Computing Systems Laboratory, division at ICS/FORTH, Antonis Papadogiannakis, Giorgos Vasiliadis, Alexandros Kapravelos, Andreas Sfakianakis, Zaxarias Tzermias, Thanasis Petsas, Eleni Gessiou, Spyros Ligouras, Giorgos Kondaxis, Dimitris Antoniadis, Elias Athanasopoulos, Iasonas Polakis, Michalis Polychronakis, Spiros Antonatos, Christos Papachristos, Manolis Stamatogiannakis, Eirini Charitaki and Meltini Christodoulaki, that contributed for a pleasant and productive environment over the last three years in the lab.

I am particularly grateful to Prof. Christopher Kruegel and Prof. Giovanni Vigna for being my mentors during my summer internship in the University of California, Santa Barbara and for the flawless cooperation we had. I would also like to thank all the members of the UCSB's Security Lab for the unforgettable moments we spent. Thank you guys!

My best thanks to all those friends I got into this unique island. Even I do not mention them by name, in the fear of forgetting some of them, I would like to thank them sincerely for their support and for sharing with me all these years of my life.

I would like to express my deepest gratitude to my parents, Nikos and Maria, my lovable little sister, Nantia, and my grandparents for their support, patience, encouragement and wise advice during my whole life. Truly, I never would have

made it through, without their love and understanding.

Finally, I would like to thank all those who believed in me, from the time of my birth until now.

To my family

Contents

1 Introduction **1**

2 Background **5**

2.1 Terminology 5

2.2 Structure of an Ad Exchange 8

2.3 How the Auction Process Works 9

2.4 How Arbitrage Works 10

2.5 Known Types of Fraud 11

2.6 Known Types of Attacks 12

2.7 Known Methods of Detection and Prevention 16

2.8 Botnet-Related Ad Fraud: A Case Study 18

2.9 Challenges in Detecting Fraud 21

3 Data Collection **23**

3.1 Data Feed 23

3.2 Establishing Baseline Traffic Patterns 25

3.3 Classifier Warnings 27

3.4 Pitfalls of Reverse Spidering 28

4 Fraud in RightMedia **31**

4.1 Cookie Replay Attacks 31

4.1.1 Clicks from the Cloud 33
4.2 Spoofing the Referrer 33
4.3 Unrecognized Referrers 34
4.3.1 Missing-In-Action Sites 35
4.3.2 Spoofing Section IDs 35
4.3.3 The Blank Referrer Problem 36
4.4 Malicious Publishers 38
4.5 Fake Web Sites 39

5 Countermeasures **41**
5.1 Digital Signatures 41
5.2 Consistency vs. Flexibility 41

6 Related Work **43**

7 Conclusions **45**

List of Figures

2.1 An overview of an online ad exchange. 8
2.2 Sample HTML iframe that instructs the web browser to download an advertisement. 9
2.3 Number of fraudulent impressions and clicks from a click-fraud botnet. 20
2.4 Configuration file for a malware program that exploits the Right-Media ad exchange. 22

3.1 Number of impressions, clicks and conversions over one month period. 28
3.2 Classifier warnings for the top 100 publishers. 29

List of Tables

3.1 Statistics for each traffic flow. 27
3.2 Breakdown of traffic by User-Agent. 27
3.3 Statistics about the types of warnings we saw. 28

4.1 Suspicious cookie statistics for each traffic flow. 33
4.2 Breakdown of suspicious cookie traffic by User-Agent. 33
4.3 Top 10 publishers using unknown referrers for April 2011. 36
4.4 Top 10 unknown referrers for April 2011. 37

Chapter 1

Introduction

Online advertising has developed into a massive economy and is now the main source of revenue for some of the most popular online businesses and search engines (like Google and Yahoo) [1]. In its simplest form, online advertising is a buyer-seller relationship between those who want to show ads (advertisers, who buy space on web pages) and those who get paid to display ads for others for a fee (publishers, or sellers, who own the web pages). The process becomes more complicated as more advertisers and publishers are added to the system. To facilitate these endeavors, an intermediary entity called an ad network (or ad commissioner [2]) keeps track of the publishers and advertisers within its domain. It is the ad network's job to take the publishers' ad requests (which are generated when users load the publishers' web site) and pair them with the most profitable advertiser payouts that matches what content that the publisher wants to display. In turn, the network takes a percentage of all revenue that is exchanged in transactions that it oversees. Thus, in the network model of advertising, the buyer/seller relationship is replaced with a broker process that seeks to maximize revenue for all involved parties.

An ad exchange, such as Google's DoubleClick or Yahoo's RightMedia, operates similar to an ad network, only the entities in an exchange are ad networks who

have their own advertisers and publishers. This allows one ad network to sell its publishers' ad space to another network or buy ad space for its advertisers. Unlike an ad network, an exchange is not fully connected, and networks in the exchange cannot buy and sell each other's traffic freely until they have established a contractual agreement.

While ad exchanges provide a powerful mechanism for advertisers, ad networks, and publishers to efficiently manage their ad traffic, they have also become a lucrative target for cybercriminals. In particular, miscreants have developed malware that is used to remotely control compromised computers, known as *bots*, and network them into a single *botnet*. A botnet provides many advantages to cybercriminals who target online ad exchanges. Most importantly, a botnet typically consists of many compromised computers with a large degree of geographic and software (browser) diversity that can be instructed to view and click ads. It is also possible that they can be directed to fill in form values on ad landing pages to further impersonate legitimate user activities and to earn additional profits. As a result, a botnet operator can generate revenue simply by creating a web site, signing up as a publisher, and directing his bots to view and click on the advertisements contained on his own web site.

In this thesis, we collaborated with a large ad network that is a member of Yahoo's RightMedia, one of the largest ad exchanges. This enabled us to obtain a direct view of how a large real-world ad network operates in the context of an ad exchange, and to analyze the security threats and fraudulent activities that pose the greatest risks. Due to the sensitive nature of the subject, we will refer to this ad network throughout the paper as *NetworkX*. In addition, we obtained access to a command-and-control server that was used to control a botnet that engaged in ad fraud targeted towards the RightMedia exchange. To the best of our knowledge, this is the first large-scale study of fraudulent activities in online ad exchanges from

these vantage points.

The remainder of this paper is structured as follows. Chapter 2 describes the structure of an ad exchange, as well as known types of fraud and methods of detection. Chapter 3 presents the dataset that we utilized to study the RightMedia ad exchange. Chapter 4 examines anomalies that indicate fraudulent activities. Chapter 5 proposes countermeasures to combat the various forms of fraud. Chapter 6 surveys related work, and Chapter 7 concludes the paper.

Chapter 2

Background

We begin by introducing relevant background information about online advertising in an exchange and documented types of fraud as they pertain to the ad exchange. In Section 2.1, we introduce common online advertising terms as they apply to RightMedia. In Sections 2.2, 2.3, and 2.4 we will describe how ads are commonly served in RightMedia. Then, Sections 2.5, 2.6, and 2.7 outline the documented types of fraud that can be perpetrated in the exchange. Lastly, in Section 2.8, we look at a case study of fraud in an exchange to get an idea of how these attacks are performed in the real world.

2.1 Terminology

Below are definitions of the various advertising terms we will use in this paper. Note that some of the terms below are defined in the context of Yahoo's RightMedia.

- *Publishers* make money through the exchange by hosting web sites with advertisements. The more visitors they attract to the web sites, the more money they earn.
- *Advertisers* pay into the exchange in order to have their ads displayed on

publishers' web sites. The more that their ads are shown, the more they have to pay to the ad network, of which a percentage is paid to the publisher.

- *Ad Networks* are entities in the exchange that manage publishers and advertisers. They are able to buy and sell traffic internally as well as through other ad networks. Ad networks that can buy and sell traffic between each other are called *affiliates*, and each ad network maintains its own list of trusted affiliated networks.

- A *creative* refers to the content of the actual advertisement, which is what the visitor sees on the page after the ad is served. The ad normally consists of an image or an Adobe Flash animation.

- Instead of having static ads that display the same content, publishers load ads dynamically by putting *sections* (also called *zones* or *regions*) on their pages. A section simply refers to a block of space on the page that is able to make a request for an ad dynamically when the page is loaded. In practice, this is normally implemented by embedding an iframe that loads some Javascript in the page that detects the browser's presence of Adobe Flash and whether browser cookies are enabled.

- The *auction process* refers to how a section is populated by a creative. It involves matching up a *seller line item*, which is essentially an ad request, with the most profitable *buyer line item*, or bid on the request. Buyer and seller line items are matched autonomously in the exchange based on the advertisers' and publishers' marketing goals. A single successful auction in the exchange is called an *impression*.

- A *click* event is generated when a user clicks on an ad, and usually brings more revenue to the publisher than an impression alone. Clicks and impressions

are handled separately in the exchange, so a user loading a page and clicking on an ad actually generates two events, an impression and a click. The percentage that an ad is clicked after being served is the ad's *Click Through Rate* (CTR). Publishers' CTRs are also recorded for use in fraud detection.

- *Ad campaigns* are the way that advertisers specify how much they pay out when their ads are shown. There are many different types of campaigns, but the most common type are based on *Cost per Mille* (CPM) impressions. In this scheme, an advertiser pays an amount to the publisher for each ad that gets served to the site.

- Additional ad campaigns types are *Cost per Click* (CPC) and *Cost per Action* (CPA). CPC deals pay the publisher only when a user clicks the ad that is served; CPA deals only pay the publisher when a user clicks the ad and continues on to do some action on the site (known as a *conversion*), usually filling in a landing page form. Because the amount of revenue associated with CPC and CPA deals depends on a user clicking the ad, the server estimates how much the ad will pay by calculating the *effective Cost Per Mille* (eCPM) impressions. The exact formula for this is: eCPM = (Payout per impression) + (Historical CTR) * (Payout per click) + (Historical actions to impressions) * (Payout per action).

- In addition to the auction process, there is a practice called *arbitrage* that ad networks can use to increase their revenue. Arbitrage is the practice of ad networks buying impressions from publishers as if they were a real advertiser, and starting a new auction for the ad slot as if they were a real publisher. As we will discuss later, this had a number of implications that had to be accounted for when analyzing the data stream from NetworkX.

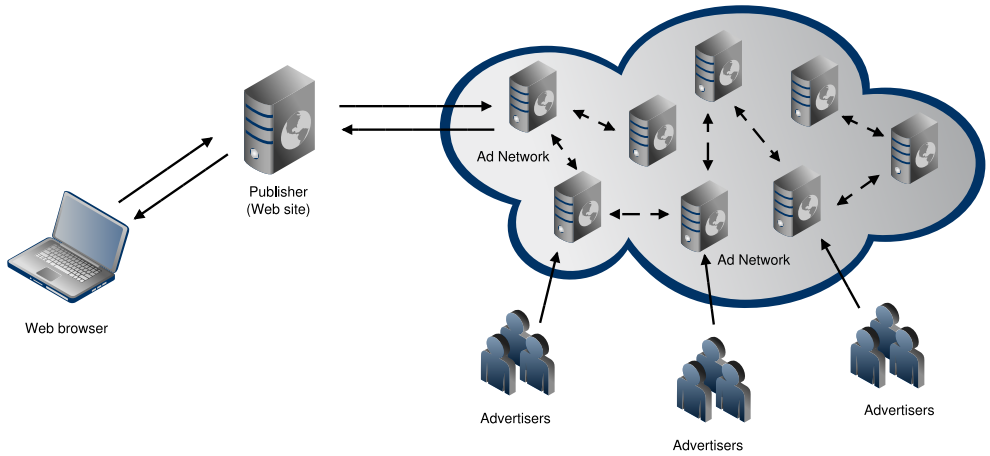

Figure 2.1: An overview of an online ad exchange.

2.2 Structure of an Ad Exchange

The exchange is structured as a graph with each node being either a publisher, advertiser, or ad network. Publishers are only able to request ads (sell traffic) and advertisers can only bid on requests (buy traffic), but networks are able to both buy and sell traffic, allowing them to act as brokers between different parts of the exchange that would not be connected otherwise. Thus, publishers and advertisers are edge nodes and ad networks form the backbone of the exchange as shown in Figure 2.1.

As previously mentioned, the exchange is not fully connected and traffic cannot be bought and sold freely across the exchange. This is because an entity can only interact with another entity if they are affiliates. Affiliations define an edge in the graph; when two entities become affiliates they decide on the specific ad campaign that will define their relationship in the exchange. This relationship is most commonly a revenue share deal. For example, NetworkX may take 50% of all publisher revenue for a particular affiliate, who targets traffic from users interested in German news, and 60% of publisher revenue with another affiliate that targets American sports traffic.

```
<IFRAME WIDTH=728 HEIGHT=90 SRC=http://ad.mynetwork.com/
st?ad_type=iframe&ad_size=728x90&section=4497316></IFRAME>
```

Figure 2.2: Sample HTML iframe that instructs the web browser to download an advertisement.

2.3 How the Auction Process Works

In the ad network model of online ad serving, one advertising network has access to a number of advertisers and publishers and the network searches for which advertisers and publishers would be most profitable together. The ad exchange replaces this with an auction process that starts when a user loads a web page that contains an HTML iframe or JavaScript popup that embeds a request for an advertisement of a specific size. An example of an HTML iframe that loads an advertisement is displayed in Figure 2.2. As the example shows, a unique identifier, known as a *section ID*, enables an ad network to track impressions, clicks, and conversions from a particular publisher. Note that the section ID is the only information that an ad network uses to determine the publisher that should be credited. As we will discuss later, this makes verifying the legitimacy of traffic difficult.

The iframe contained on the web site that contains the URL with a section ID is resolved to the corresponding publisher by the ad exchange. This section ID is associated with a seller line item ID, which contains a number of parameters regarding the transaction. The seller line item cascades through the exchange until an advertiser receives the request. The seller line item has information associated with it, such as the requesting URL and targeting information. If the advertiser meets the publisher's targeting goals and the advertiser trusts the publisher, then the advertiser generates a buyer line item ID that identifies the advertiser's bid on the ad request. The actual bid amount is determined automatically by the exchange based on the advertiser's Return on Investment (ROI) goals, and is always in the

form of the effective cost per mille (eCPM) impressions, regardless of the types of affiliations between intermediary ad networks. Buyer line items are propagated back to the publisher similar to the seller line items, but the associated revenue with the bid changes at each hop. For example, assume a publisher and advertiser were separated by an ad network. The advertiser's campaign with the network is a $1 CPM deal while the publisher has a 50% revenue share deal. This means the advertiser's bids will only be worth $0.50 to the publisher, because the network is taking half as a broker. In reality, it would be even less than $0.50 because the ad exchange would take a cut as well. When the publisher receives all the bids, the exchange simply picks the buyer line item with the highest eCPM for the publisher and that ad is served.

2.4 How Arbitrage Works

Arbitrage occurs after the auction process when an ad network buys a publisher's ad request. In order to initiate arbitrage, the network must buy the publisher's ad traffic themselves and then resell the traffic in a completely new, independent auction. This is done by serving a brand new ad tag to the publisher after the initial auction. The new ad tag contains a new section ID that is owned by the ad network and not by an actual publisher. Whatever revenue is generated in the new auction goes to the ad network that won the first auction, in the hopes that the second auction will make a profit larger than the ad request cost. Because the network must buy the traffic themselves, unsold arbitrage traffic is a direct loss for the ad network. Arbitrage can be repeated a number of times across different ad networks until an actual ad is served; this process is called *daisy chaining*. The longer the chain of arbitrage, the longer it takes to load the ad that is finally displayed.

2.5 Known Types of Fraud

Fraud (or *Ad Fraud*) in the context of an ad exchange is a means by which a member or members in the exchange try to increase their profits at the cost of other members in the exchange. A *fraudster* is simply a member of the exchange who is perpetrating fraud of some kind. The fraudster may be a publisher in the exchange who is attempting to make more money than they deserve, or an advertiser who is targeting other advertisers to reduce ad competition. Below are the known and documented types of fraud that may be perpetrated in the context of an ad exchange:

- *Impression Spam*: Impression spam is the simplest type of ad fraud, and it involves fabricating HTTP requests to either the publisher's page, or the ad server directly, in order to inflate the actual amount of traffic. This type of fraud targets CPM deals, but may be mixed in with other types of fraud in order to remain stealthy [3]. Impression spam can be difficult for advertisers to perform on other advertisers, simply because an attacker does not usually have control of the ads that will be shown on a particular site. This means there is a chance the advertiser will be performing fraud against themselves, unless they can ensure that only their competitor's ads will be shown.

- *Click Spam*: Click spam involves making HTTP requests to advertisement click URLs when the clicker of the ad is not really interested in its content. There are two kinds of click spam fraud. *Click inflation* is the practice of publishers making more money than they deserve through inflating CPC deals, or increasing their CTR and thus their eCPM. *Competitor clicking* is the practice of advertisers generating HTTP requests to competitor advertisers' ad URLs to deplete their advertising budget. In addition, competitor clicking drives down the competitor's ROI for certain targeting parameters, thus

lowering the cost of future legitimate clicks for the fraudster [3].

- *Conversion (Action) Spam*: Fabricated HTTP requests to a specific advertiser-defined URL that requires certain GET or POST parameters, requests a file for download, or follows a specific order of pages in order to generate a conversion against that advertiser. Like click spam, this can be perpetrated by either publishers or advertisers. This type of fraud only works if the action does not require spending money directly, such as purchasing an item from the web site.

- *Misrepresentation*: The practice of a publisher breaking some rule of the network or exchange by lying about their web site content or lying about what pages ads are actually being shown on (for example by spoofing the referring URL). The publisher normally does this to get higher value ads on their pages than they would be able to get if they did not lie about their website content, or because their web site content is illegal and would not be allowed in the network otherwise.

2.6 Known Types of Attacks

Below are the known types of attacks that either have been performed or could be performed in the context of an ad exchange:

- *Hired Clickers*: This type of attack involves someone sitting in front of a computer and constantly reloading a page and clicking on ads. It is limited by the fact that humans cannot work as fast as a computer and by the fact that laborers must be paid in order to perpetrate this attack on a large-scale. To avoid detection from having a static machine and IP, this type of fraud is often obfuscated by using a proxy or list of proxies that the perpetrator

rotates through. Although this type of fraud is less profitable than automatic fraud because of human overhead, it can be much harder to detect, especially when fraudulent traffic is mixed in with legitimate traffic. In addition, there are "click farms" available for hire that will generate fraudulent traffic for you; these are often located in third- world countries to reduce labor costs [3].

- *Pay-to Sites*: A type of hired clicker attack where web sites owned by a fraudster offer a "pay-to-read" or "pay-to-click" deal to any users who are willing to sign up. Once the user signs up, the site pays the user a small stipend to read target pages with large numbers of ads, often times with specific instructions to "stay on the page for at least 30 seconds and click on whatever looks interesting." Because the fraudsters own both the pay-to sites and the target pages, the fraudsters are able to get human traffic without having to generate real content to attract viewers. Interestingly, these "pay-to" sites often advertise themselves on other "pay-to" sites, after all, advertising is all about targeting.

- *Keyword Stuffing*: A type of misrepresentation fraud that increases the value of ads that are shown on the fraudster's pages. This is done by including a certain amount of "invisible" content that contains many high-value advertising keywords. The invisible content is either in hidden HTML tags, text that is the same color as the background, or very small text. When the network crawls the fraudster's page in order to classify the content, the page will be classified as being more valuable or targeted than it really is. This drives higher value ads to the fraudster's page, who can also choose to include content that drives many users to their site, such as porn or game sites. In this way, the fraudster gets the advertisers to pay out more for less valuable traffic. It is worth noting that if the ad network is small and classifies its

publishers manually, than this type of fraud will not bring any benefit to the fraudster.

- *Impression Stuffing*: The practice of fraudster's putting excessive numbers of banners on their pages so that they get a large number of impressions for each page view. This also includes "stacking" ads on top of each other so background ads cannot be seen. In order to bring traffic, the fraudster must disseminate links to their site, which may be located on public forums, like 4chan, or they may be propagated via compromised social network accounts [4].

- *Coercion*: This attack is perpetrated by fraudsters who convince users to click on their ads for reasons other than the ad content itself. This includes an administrator simply asking users to click on their ads, but also includes obfuscating ads with actual site content in order to trick a user into clicking on ads (for example, making all valid links on the page look like ads as well). This type of attack harms both advertisers who are not getting genuine clicks and users who often get confused and redirected to a page they did not want to be on.

- *Custom Clickbots*: Custom-made software by fraudsters that perpetrates a particular kind of fraud against certain publishers or advertisers. These clickbots normally sit on one or more static machines and issue HTTP requests to certain URLs in order to simulate impressions, clicks, or conversions. These types of bots can be easy to detect when they are not designed by someone who knows how fraudulent traffic is detected, but do not have known behavioral patterns like the for-sale clickbots [3].

- *For-sale Clickbots*: Software that is available for download or purchase that will perform click fraud. These bots are configured to perform many types

of fraud and can be given lists of publishers' pages to visit, ads to click on, and proxies to use to diversify the bot's IPs. However, because they are sold commercially, anyone (including fraud researchers) can purchase and study the bot's behavioral patterns, potentially making detection easier.

- *Botnet Clickbots*: This is the most difficult type of clickbot to detect from an ad exchange's perspective, and it is the most common source of fraudulent Internet ad traffic [5]. These come in two flavors: ones that run behind the scenes and act as a normal clickbot and those that attempt to coerce the user of the machine to perform some of the ad fraud themselves. In the former case, the malware could simply be a for-sale clickbot that was installed on a user's machine via a Trojan and can be controlled by the botmaster. If the malware is attempting to coerce the user, it may display popups in the hopes that the user might click on them and even go on to generate a conversion [6]. Botnet malware can also be used to replace legitimate ads in a victim's browser with those of an attacker. As a result, if a user clicks on an ad link they are actually clicking on an advertisement that benefits the botnet controllers. In addition, bot malware can intercept requests to targeted sites, and redirect the victim to an affiliate web site via DNS or browser hijacking (through a browser helper object or extension).If the user makes a purchase through the affiliate site, the botmasters typically receive a percentage of the transaction.

- *Forced Browser Clicks*: An attack that forces a user's browser to follow the click URL of an ad by including some client-side script, normally Javascript. This type of attack is commonly avoided by putting all ads in an iframe with the source attribute set to an ad tag located on the ad network's web site. By using this technique, the content of the iframe is not accessible to

any script that did not come from the same domain as the iframe's source. However, research done at the University of Indiana suggests that there are still ways of getting click URLs out of iframes. One such method involves blindly crawling the source attribute and recursively following hyperlinks to get a list of possible click URLs. Once the URLs are collected the script is then able to redirect the user to one of these links at random, thus generating the fraudulent click [7].

2.7 Known Methods of Detection and Prevention

Below are the known defenses against ad fraud in the context of an ad exchange.

- *Rule-based Detection*: This type of detection uses static rules to decide what traffic should be counted and what is invalid. One example of a common rule is that the second of any duplicate clicks (caused from a user double-clicking an ad) is considered invalid. This type of detection is beneficial in finding known attacks by looking for known malicious patterns, but it does not work on attacks whose patterns are not known or do not follow static rules [8].

- *Classifier-based Detection*: This uses anomaly-based methods and historical information about every publisher to find sudden changes in ad traffic patterns. This may involve looking for a sudden increase in the number of impressions from a publisher, the CTR of the publisher, or the classification/quality of traffic the publisher is generating (for example if the publisher is a search engine that suddenly only queries for high-value ad keywords). This type of detection is useful for identifying publishers who are misbehaving or for when fraudsters change or update the type of attack they are perpetrating.

- *Reverse Spidering (Auditing)*: The practice of ad networks, ad exchanges, or advertisers crawling the HTTP referrer of incoming impressions in order to

ensure the referring sites have the content they claim they do. The reverse spiders look at keywords in HTML content, Javascript, and iframes of the pages to look for any potentially illegitimate content. To avoid this kind of detection, fraudsters assign a unique ID in the referring URL each time a fraudulent click or impression is generated. This way, when the audit program crawls the referrer, the web site will recognize that the ID has already been used and not serve any malicious content to the spider [7].

- *Bluff Ads*: Bluff ads are ads that an ad network serves to a publisher in order to detect fraudulent activity. These ads are served to a random percentage of all requests that come from the publisher and are unique in that they are purposefully uninviting (meaning they contain little more than a picture with no text that does not try to attract the user's attention or get them to click on it). If the click through rate and conversion rate of these bluff ads is not much lower than ads normally displayed, this would indicate fraud from the publisher [9].

- *Web Site Popularity and Page Rankings*: The number of impressions a publisher is generating for their web page can be checked against known, trusted web site rankings such as Alexa or Compete. If the publisher has much more traffic than their page ranking would suggest, this would be indicative of fraudulent activity [10].

- *Performance-based Pricing*: Performance based pricing is simply a name for publisher payment schemes that do not pay on impressions but instead on how much ROI an advertiser gets from a publisher. The simplest performance-based pricing model is CPC, but CPA deals fall into this category too. This type of pricing reduces impression fraud by requiring publishers to provide a certain level of measurable benefit to the advertisers in order to make money

(however this can still be spoofed, for example, click fraud). This also reduces cost to advertisers as networks with more fraud will have a lower ROI for their advertisers, meaning that advertisers will have to pay less to get their ads shown. In this way, performance-based pricing mitigates the effect of fraud on the advertisers without actively avoiding it [11].

2.8 Botnet-Related Ad Fraud: A Case Study

In this section, we describe the process that facilitated our efforts in obtaining access to a command-and-control (C&C) server that controlled a botnet used to commit ad fraud, and the data that we collected. The bot malware first came to our attention in February 2010, when we discovered a suspicious binary sample that exhibited peculiar behavior. The sample was submitted to ANUBIS [12], an automated system that executes a Windows program in a virtual environment and records its actions (e.g., file system modifications, process creation, and network activity).

By analyzing the network connections generated by the malware sample, we were able to identify the location of the C&C server that was providing instructions to the bot. We then contacted the hosting provider whose server was used for controlling the operation and provided them with detailed evidence of the abuse. The hosting provider suspended the criminal account in March 2010 and provided us with access to the information stored on the server. Note that we had previously established contact and collaborated with this vigilant ISP through FIRE [13], our network reputation service that tracks where malicious content is hosted on the Internet.

By studying the behavior of a bot sample and the source code of the botnet C&C, we were able to get a complete view of how the entire operation functioned. There were two primary methods the botmasters used to earn money: impres-

sion/click fraud and affiliate programs that paid a commission based on conversions (e.g., registration, sales, etc). The mode of operation for the impression/click fraud was managed by a configuration file received from the C&C server as shown in Figure 2.4. The configuration contained various parameters for controlling the patterns of impressions and clicks, iframes to load within the web browser, a blacklist, and a list of domains that were used to spoof the source of the impression/click through the HTTP referrer field. Note that the end of the configuration file contains a fake HTML page to mimic a publisher's web site. The iframes are directly loaded by the malware in the background and are invisible to the user of the infected system. Thus, the malware does not need to visit an actual web site to emulate impression and clicks. This way, the fraudsters who own the web pages that the section IDs are supposed to be displayed on do not have to pay for the botnet's bandwidth to these pages.

Interestingly, we found that the third-party domains used to spoof the origin of the traffic contained RightMedia ads. We will discuss the purpose of spoofing the referrer later in Section 4.2. There were also a small number of domains (and web sites) in the configuration file that were set up and used to register multiple publisher accounts that at first glance appeared somewhat legitimate. On further examination, we identified that the content on these pages was actually stolen from other sites. The format of these sites were identical, namely a Wordpress blog template, with posts only by an "admin" user, no comments, and a large number of ads (from several different ad exchanges) embedded throughout each of the pages on these sites. We will revisit these fake sites in Section 4.5.

Periodically, the infected computer would connect back to the C&C server to report its status, and receive a new list of instructions. We also found that the C&C server had the ability to push arbitrary binary executables to the bots; this was regularly used to upgrade the click-fraud malware to newer versions. The

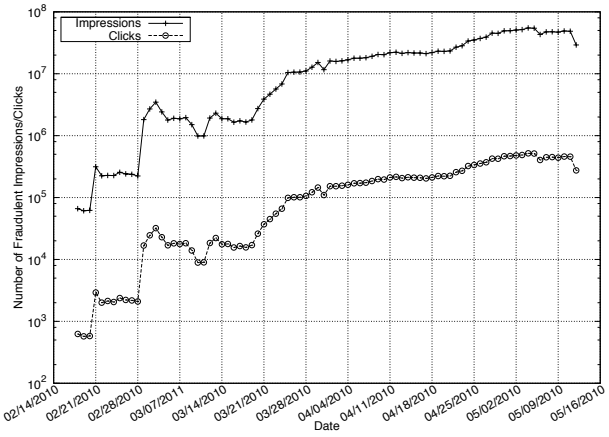

Figure 2.3: Number of fraudulent impressions and clicks from a click-fraud botnet.

bot malware also used browser hijacking to redirect users from a target site (e.g., freecreditreport.com) to an affiliate site (e.g. incentaclick.com and adsmarket.com). Depending on the referring site, the affiliate site would redirect a user's browser to a similar web site (e.g., gofreecredit.com). Based on the records maintained by the botnet controllers, their malware redirected users 3,425,030 times from mid-February to May 2010. However, the number of conversions was not stored on the C&C server so we cannot determine how many users fell for the scam.

The database for this particular botnet contained records for 530,985 bot installations, with 1,479,036,685 impressions and 14,060,716 clicks (an average click through rate of 0.95%) from mid-February to May 2010. The daily number of impressions and clicks is shown in Figure 2.3. Interestingly, the ratio of impressions to clicks followed each other almost exactly over time. As we will discuss later in Section 3.2, the average CPM for impressions and cost-per-click were $0.084 and $0.017, respectively. In other words, the cybercriminals behind the botnet may have netted approximately $125,940 for impressions and another $255,000 for clicks during this 2.5 month period.

We would also like to point out that this botnet malware operates based on an affiliate program based out of Eastern Europe (commonly referred to as

a *partnerka*) similar to other online criminal operations such as fake antivirus ventures [14]. More specifically, attackers are paid to compromise as many computers as possible and infect them with malware. It should be noted that this botnet has been operational for more than a year, and is still currently active.

2.9 Challenges in Detecting Fraud

Whether intentional or by design, RightMedia does not provide a strong sense of accountability among ad networks. Instead much of the responsibility relies on the ad networks to monitor for suspicious behavior. In addition, there are only a limited number of options that are provided to an ad network to detect fraud. For example, many of the data fields in the ad traffic field that RightMedia provides are simply not populated for any brokered traffic. Further, some fields can be seletively supressed by RightMedia, which caused us problems in identifying potential fraudulent publishers, as we will discuss in Section 4.3.3.

```
<xml id='timestamp'>1305159959</xml>
<xml id='refresh_time'>90</xml>
<xml id='feeds_amount'>11</xml>
<xml id='max_impress'>2</xml>
<xml id='glob_click_limit'>2</xml>
<xml id='refresh_time_zero_factor'>30</xml>
<xml id='click_interval'>1800</xml>
<xml id='referer'></xml>
<xml id='refresh_timeout'>90</xml>
<xml id='cookie_cleaning_enabled'>false</xml>
<xml id='cookie_cleaning_interval'>25</xml>
<xml id='cookie_domains'></xml>
<xml id='redirects_enabled'>true</xml>
<xml id='redirects_db_timestamp'>1305066552</xml>
<xml id='domains_blacklist_enabled'>true</xml>
...
<html>
<head><title>Error</title></head>
<body>
<iframe name='1224' provider='616' clicklimit='1'
    src='ads/160x600/4bed5c321af24.html' width='160'
    height='600' feed_cap='5' scrolling='no'></iframe>
<iframe name='198' provider='368' clicklimit='1'
    src='ads/300x250/4bed5c31efd4d.html' width='300'
    height='250' scrolling='no'></iframe><br/>
<iframe name='784' provider='521' clicklimit='1'
    src='ads/728x90/4bed5c320ee0e.html' width='728'
    height='90' feed_cap='8' scrolling='no'></iframe>
</body>
</html>
```

Figure 2.4: Configuration file for a malware program that exploits the RightMedia ad exchange.

Chapter 3

Data Collection

In this Chapter, we present the dataset that we utilized to study the RightMedia ad exchange.

3.1 Data Feed

We applied our system to real-world ad data from NetworkX, which is an ad network that is part of RightMedia's exchange. We received new data every 30 minutes, and, therefore, we were able to apply our system to fairly recent ad data. The traffic can be split into three distinct flows based on the different types of transactions that are allowed in RightMedia. These flows are local publisher traffic, arbitrage traffic, and auction traffic. Local publisher traffic is any traffic that originated from an ad request from one of NetworkX's own publishers. Arbitrage traffic is traffic that was either purchased by or sold from NetworkX itself. Arbitrage traffic comes in pairs, with one impression being for purchasing the ad traffic and a corresponding one for reselling the ad traffic. Auction traffic is any traffic that NetworkX made money on through affiliations as a middleman, but NetworkX did not buy or sell directly. Depending on which flow we were analyzing, different amounts of data were available to us.

In order to study the data feed, we implemented an automated system to periodically retrieve NetworkX's data records and extract relevant information. The data itself was in the form of Right Media's *Custom Data Feed* format, which contains detailed information for all impressions, clicks, conversions as well as for auctions and arbitrage. More specifically, we examined the following data fields:

- *IP Address*: Right Media only provides an ad network with the first three bytes of the IPv4 address, in order to preserve users' privacy. However, we were able to distinguish how many different users were in a class C network based on how many unique cookie IDs we identified per IP.

- *Cookie ID*: Unique token given to each person who views an ad; it is stored as a cookie on the local machine and sent with every ad request. Note that this ID is a hash of the actual cookie value.

- *Creative ID*: Unique ID given to each creative in order to track which ad was shown specifically.

- *Section ID*: Unique ID for the particular ad space where this ad was shown. For arbitrage traffic, this corresponds to a section ID that NetworkX owns, and for local traffic, it corresponds to a section ID that a local publisher owns. This field is not populated for auction traffic.

- *Referrer*: web site URL of the referring web site for local publisher traffic, or a subdomain of NetworkX for arbitrage traffic. Not populated for auction traffic.

- *Impression/Click*: Whether it was an impression or click.

- *Advertiser/Publisher Revenue*: How much the publisher got paid for the ad and how much the advertiser paid out. The difference between the advertiser

cost and publisher revenue is how much the intermediary ad networks earned for the ad.

- *Buyer ID*: Unique ID given to each advertiser.
- *User Agent ID*: Identifies the user-agent field that was specified in the HTTP headers. RightMedia currently enumerates 40 different types of browsers and versions.
- *Region ID*: Identifies local geographical areas (i.e., a state, province, or city) based on IP-based geolocation services.

In addition, we had access to the seller ID (publisher ID), as well the associated line item IDs, but we did not use these fields in our analysis because they cannot be manipulated by fraudsters, and thus are not a good indicator of potential fraudulent activities.

3.2 Establishing Baseline Traffic Patterns

In order to analyze the traffic for anomalous activity, we first collected general statistics about the data in order characterize its traffic patterns. Overall statistics for each traffic flow are outlined in Table 3.1. We performed most of our analysis on the local publisher traffic, because this data set contained the most information about each impression. Table 3.2 has a breakdown of publisher traffic by user-agent. We will reference these tables later as a baseline for expected traffic patterns and compare this with observed traffic patterns in likely fraudulent incidents.

To establish thresholds for our experiments, we observed the impressions, clicks, and conversions from the publisher data flow for April 2011. On average, we measured a click- through-rate (CTR) of 0.3%, with conversions being 3.33% of the total amount of clicks, as shown in Figure 3.1. In addition, we measured an average

CPM of $0.084, an average cost-per-click (CPC) of $0.017, and an average cost-per-action (CPA) of $0.545. From these results, we developed a classifier to analyze particular aspects of the local publisher dataset. The data analyzer collected, for each cookie ID and IP subnet, the number of impressions, unique IPv4 (number of class C) subnets, unique cookie IDs, revenue paid to the publisher, click-through rate, user-agent and region distributions, and total numbers of buyers (advertisers) and sellers (publishers). This was designed to be strictly an *offline* analyzer (classifier), in that it ran on historical data only and not real-time data. We used the following thresholds to establish what patterns we considered fraudulent, as measured over a one hour moving window. These thresholds are all set at three standard deviations above the mean:

IP thresholds:

- Impressions for a single IP >32
- CTR for a single IP >0.2%
- Total publisher revenue for a single IP >$0.12

Cookie thresholds:

- Impressions for a single cookie >20
- CTR for a single cookie >0.3%
- Total publisher revenue for a single cookie >$0.08
- Number of IPs for a single cookie >10

We were able to analyze 39 days of traffic, from March 28, 2011 to May 6, 2011. The most interesting results came from analyzing the local publisher traffic, of which we processed 38,378,034 impressions for analysis. In addition, we processed

605,098,102 impressions from the entire traffic stream into a separate database, in order to analyze inter-network traffic. In particular, we were looking for any IPs or cookies that consistently violated our thresholds. Such violations conceivably indicate fraud.

Traffic Flow	**Traffic (%)**	**Impressions (per hour)**	**CTR**	**Conversion Rate**
Auction	91.4%	4,392,382	0.3%	-
Publishers	6.6%	315,174	0.2%	0.01%
Arbitrage	2.1%	99,018	0.4%	0.007%

Table 3.1: Statistics for each traffic flow.

Browser	**Impressions**	**Traffic (%)**
IE 8.0	10,946,023	28.5216
Internet Explorer 6.0 - Windows	5,726,732	14.9219
Firefox 3 - Windows	5,369,066	13.9899
Google Chrome	4,497,019	11.7177
Internet Explorer 7.0 - Windows	4,173,819	10.8755
Mozilla - Other	2,547,383	6.6376
Safari 2 - Mac	872,760	2.2741
Fun Web Products (any browser)	847,833	2.2092
Mozilla - Windows	572,815	1.4926
Opera - Windows	492,712	1.2838

Table 3.2: Breakdown of traffic by User-Agent.

3.3 Classifier Warnings

Over the course of the 39 days of data that we analyzed, our classifier generated 2.3 million warnings for different cookie IDs and IP subnets. Figure 3.2 shows that a vast majority of warnings come from a small subset of publishers. More specifically, three publishers accounted for 50.5% of warnings and the top 20 produced 87.4% of all warnings. Thus, these publishers that consistently produced warnings justified further investigation.

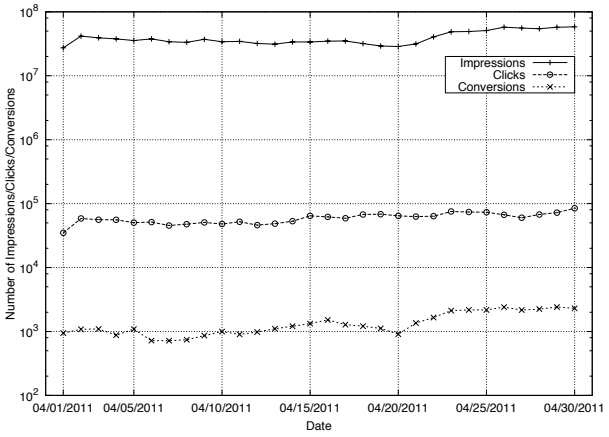

Figure 3.1: Number of impressions, clicks and conversions over one month period.

IP subnets can be attributed to 37% of these warnings. In particular, we were interested in cookies and IPs that consistently violated our thresholds and thus generated many warnings. Our logic was that if an IP or cookie was being used for fraud, the perpetrator would have to be using it consistently in order for the fraud to be worthwhile. Table 3.3 shows that a majority of incidents did not violate our thresholds consistently, and thus did not raise any alarms. The incidents that violated our thresholds over the course of 39 days were most likely to be fraudulent such as the cookie is described in Section 4.1.

Type of warning	**Percent of warnings**	**Number of incidents**	**Incidents >1 day**
Cookies	63.1%	1,319,344	0.008%
IP 24-bit Subnets	36.9%	187,386	1.33%

Table 3.3: Statistics about the types of warnings we saw.

3.4 Pitfalls of Reverse Spidering

In addition to the data feed analyzer, we also developed a reverse auditing system that crawled the referrers for each impression. We deployed 50 virtual machines that utilized a portable software-testing framework for web applications, known

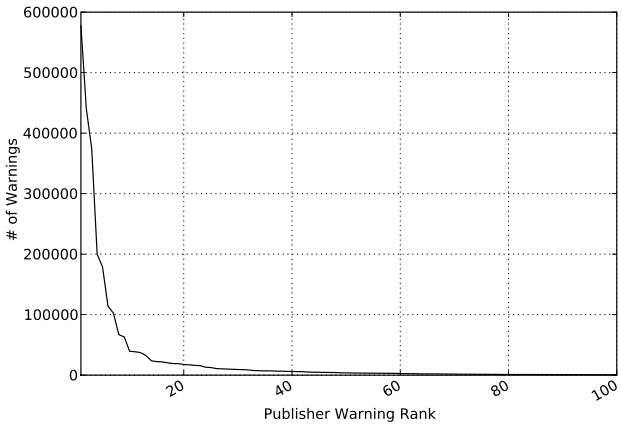

Figure 3.2: Classifier warnings for the top 100 publishers.

as SELENIUM [15], which has the ability to programmatically control a real web browser. The primary benefit that Selenium provided was the ability to interpret JavaScript code. For each audit, we ran TCPDUMP, a network recording tool to extract the network requests. From the data we obtained from the recorded requests, we extracted the section ID fields that are responsible for generating revenue for the publisher. Then, we compared the section ID values from NETWORKX's data feed with the observed section ID values found on that referrer's site. The motivation behind this process was due to the behavior of the botnet malware that we described in Section 2.8, which spoofed the referring URL with third-party sites that contained RightMedia ads while using their own section IDs.

After several months of monitoring we noticed that 79.2% of referrers in NETWORKX's data feed contained no section IDs when visited by our auditing system. Clearly, this high percentage of traffic could not be fraudulent. Therefore, we analyzed some of the referring sites that did not contain section IDs and found a common pattern. In particular, many sites appeared to use IP-based geolocation to serve particular regions different ads. In addition, some of the sites used ad de-

livery optimization services such as Casale Media that dynamically choose different ad exchanges to maximize conversions. In other words, a visitor may have received a RightMedia ad based on his location, but our crawlers, which are based in the U.S., were delivered ads from DoubleClick's exchange. As a result, we discovered that reverse spidering for the purpose of fraud detection suffers from significant limitations.

Chapter 4

Fraud in RightMedia

In this Chapter, we examine anomalies in our dataset that indicate fraudulent activities in the RightMedia ad exchange. Our offline analyzer unearthed a number of suspicious patterns that shed light on possible attacks that either could be performed or are being performed on RightMedia.

4.1 Cookie Replay Attacks

The threshold-based analysis of the data feed found one particular cookie ID that was generating a consistently large number of impressions over the entire period of our analysis. We observed this cookie's behavior in NetworkX's local traffic, in NetworkX's arbitrage traffic, and in NetworkX's auction traffic. In addition, we found numerous other irregularities related to this cookie. Normally, a cookie is associated with a single user and, thus, a single unique IP, browser, and geographic region. However, the data for the suspect cookie indicated that the cookie was coming from 28 different types of browsers, 746 global regions on 666,429 different 24-bit IP subnets, and using 28 browser languages across all the traffic. We also observed this cookie coming from 236 unique local publisher accounts. The relative distribution of these statistics deviates significantly from the rest of the traffic.

Moreover, the cookie's conversion rate is 18 times greater in Table 4.1 than for the overall traffic in Table 3.1. In addition, the distribution of operating systems in Table 4.2 is weighted significantly toward Apple's operating system, in contrast to Table 3.2, which is consistent with the current market share of operating systems (i.e., primarily Microsoft Windows).

The question of why an attacker would randomize the browser version, language, and other fields, but not refresh the cookie value is probably due to the fact that it would require writing a relatively sophisticated JavaScript parser that was capable of extracting the value that was set by the ad network. In addition, the cookies set by RightMedia are encrypted with a server-side key, and a Hash-based Message Authentication Code (HMAC) is used to verify its integrity. Thus, an attacker cannot set arbitrary cookie values without breaking RightMedia's encryption algorithm. However, it appears that cookie replay attacks are possible based on the frequency and extended period of time that we have observed this cookie in use. It also appears that the current fraud systems that are in place are not yet effective enough to detect this type of activity.

From the local publisher traffic flow, which constituted 7% of all NetworkX's traffic, we calculated the cookie was generating $32 in revenue for the fraudster per month, and costing advertisers $56 per month. Since these values were collected from only the local traffic flow, the real amount of fraud across the entire exchange would be much larger than this. To get an idea of how much the fraud scales accross the exchange, we looked at how much revenue the cookie was generating across all the traffic passing through NetworkX, and found that the cookie is generating $235 in revenue and $409 in cost every day. However, it is important to highlight the fact that NetworkX is just one of several hundred ad networks in the RightMedia exchange, so the potential loss due to fraud from this type of operation is likely far greater.

Traffic Flow	Traffic (%)	Impressions (per hour)	CTR	Conversion Rate
Auction	91.3%	27,848	0.3%	-
Publishers	7.1%	2,158	0.6%	0.185%
Arbitrage	1.6%	500	0.6%	0.114%

Table 4.1: Suspicious cookie statistics for each traffic flow.

Browser	Traffic (%)
IE 8.0	22.7588
Mozilla - Mac	14.2222
Safari 2 - Mac	11.9598
Firefox 3 - Windows	11.4521
Safari 4 - Mac	9.4828
Mozilla - Other	9.4090
Internet Explorer 6.0 - Windows	3.9338
Internet Explorer 7.0 - Windows	3.8848
Opera - Windows	3.5850
Google Chrome	3.1144

Table 4.2: Breakdown of suspicious cookie traffic by User-Agent.

4.1.1 Clicks from the Cloud

Interestingly, we identified traffic that originated from Amazon's Elastic Compute Cloud (EC2). While ad traffic from the cloud is not by itself suspicious (web users may proxy browser traffic through the cloud), we observed the cookie ID discussed previously used in a large amount of requests that originated from the cloud from April 10, 2011 to April 13, 2011. Thus, we believe that this is a strong indicator that attackers are using Amazon's cloud (possibly the free tier that allows for 30GB of transfer per month) in order to generate fraudulent ad impressions and clicks.

4.2 Spoofing the Referrer

Referrer spoofing occurs from clickbots who want to hide their fraudulent traffic across multiple referrers so that large numbers of impressions do not come from

referrers that are not ranked very highly. The bots rotate through a list of referrers while performing the fraud, but always use a section ID that the fraudster owns. We observed this type of fraud on the click fraud botnet command and control server that we had control of during our research. In February and March of 2010, we observed the command-and-control server issuing large numbers of referrers to the bots that included both the fraudster's sites and popular sites that they could not have owned, such as citibank.com. A year later, in April 2011, we observed the command-and-control server issuing only a few possible referring URLs that only included the fraudster's fake sites. We suspect the change in behavior away from spoofing the referrer, because we believe that RightMedia may have caught on to their traffic coming from a wide variety of referrers that did not match their registered publisher web sites. In the following section, we will analyze other publishers that exhibit similar behavior.

4.3 Unrecognized Referrers

Another type of analysis that we performed was looking at unrecognized referrers and why they would be generating impressions for a particular section. We define an *Unrecognized Referrer* as any referring site that does not own the section ID that the impression belongs to. In RightMedia, section IDs are allowed to be placed on sites other than the site the publisher had originally registered with, so simply observing that a section is getting impressions from unrecognized referrers is not enough to classify the impressions as fraud. Below we outline some types of attacks that we observed or that could potentially be perpetrated because of this policy. This type of analysis could only be performed on traffic from local publishers as this is the only traffic flow that has the referrer set, and we computed that unknown referrers made up 43.2% of this traffic.

4.3.1 Missing-In-Action Sites

One of the things we were interested in looking at while analyzing the data was how many sections had homepages that were down, but were still generating impressions from other referring sites. We call these sites Missing-In-Action (*MIA*) sites, and we calculated that out of NetworkX's 1600 publishers, 10% had unreachable domains and 5% were 404's. Although we gave special attention to the sites we knew were down, a number of domains were replaced with parking pages so we analyzed all the publishers for unknown referrers. The results of our analysis for April 2011 are outlined in Table 4.3a and Table 4.3b. We looked both at publishers that had a large number of impressions from unknown referrers and those that used a large variety of unknown referrers.

From these results, we observed a specific instance of a local NetworkX publisher who was performing a kind of misrepresentation fraud with his MIA site that allowed the publisher to host ads on a page that had illegal content that violated RightMedia's terms of use. First, the fraudster registered as a benign publisher, in this case PublisherC's site, and received a number of section IDs to use on their site from NetworkX, who did not find anything wrong with the site's content. Instead of placing the ad tags on the benign site, however, the fraudster placed them on a site that contains illegal content, in this case `full-free-games.com`. Because there is no check to ensure the referrer matches the page, impressions generated from `full-free-games.com` still make money for the fraudster.

4.3.2 Spoofing Section IDs

Table 4.4a describes the most popular unrecognized referrers we analyzed, sorted by impressions and by how widely used the referrer is. A number of the referrers in Table 4.4b can be explained simply by proxy traffic, however there are a number of referring domains that we were surprised had appeared on the list. Most notably,

Publisher	Unknown Referrers	Impressions
PublisherA	300	3,624,162
PublisherB	46	2,720,146
PublisherC	63	1,640,597
PublisherD	1	1,153,357
PublisherE	55	702,209
PublisherF	19	511,066
PublisherG	6	319,442
PublisherH	22	200,334
PublisherI	6	157,033
PublisherJ	1	155,809

(a) Sorted by impressions.

Publisher	Unknown Referrers	Impressions
PublisherA	300	3,624,162
PublisherK	87	6,976
PublisherC	63	1,640,597
PublisherE	55	702,209
PublisherB	46	2,720,146
PublisherL	31	146,877
PublisherM	25	3,496
PublisherN	25	3,311
PublisherH	22	200,334
PublisherF	19	511,066

(b) Sorted by unique referrers.

Table 4.3: Top 10 publishers using unknown referrers for April 2011.

the domain `pizzahut.com` appears over 20 different sections owned by different publishers in completely different countries. Because the amount of traffic is so low and not targeted toward any publisher in particular, we hypothesize that this is a way of obfuscating fraud by distributing fraudulent traffic across a number of section IDs that the fraudster may own. By doing this, the perpetrator's section IDs will be mixed in with a number of other benign section IDs, making it harder to pin down the real source of the fraud.

4.3.3 The Blank Referrer Problem

When looking at traffic containing unrecognized referrers, we paid special attention to the kinds of referring domains that were showing up for each publisher. There were a few cases where publishers had a majority of their traffic coming from blank referrers with the rest being made up of proxy and translated google content. One publisher in particular, PublisherD had no referrers set in the data feed but generated 1.1 million impressions during April 2011. To investigate why this would occur, we visited PublisherD's home page and recorded the network traffic of the background ad requests to ensure that the referrer field was being properly set in the

Domain	Unique Sections	Impressions
ReferrerA	1	2,713,095
-	317	1,856,630
full-free-games.com	1	1,623,372
ReferrerB	1	1,024,615
ReferrerC	1	698,621
ReferrerD	1	560,841
ReferrerE	1	502,598
ReferrerF	1	490,653
ReferrerG	1	441,122
ReferrerH	1	355,139

(a) Sorted by impressions.

Domain	Unique Sections	Impressions
-	317	1,856,630
translate.googleusercontent.com	90	20,807
webcache.googleusercontent.com	50	1,473
s0.2mdn.net	28	76
pizzahut.com	21	161
ReferrerI	21	91
fls.doubleclick.net	18	79
static.eu.criteo.net	15	93
5.hidemyass.com	10	68
ReferrerJ	10	36

(b) Sorted by unique sections.

Table 4.4: Top 10 unknown referrers for April 2011.

HTTP header of the ad requests to RightMedia. Once we verified that this was the case, we searched our data feed for the impressions we put through for PublisherD to see if the referrer was set. Our logic was that if the referrer was set in the feed for the impressions we generated, this would be highly indicative that PublisherD was involved in fraudulent activity. However, we found that our impressions did not have the referrer set, which means RightMedia had intentionally suppressed that field before NetworkX had access to the ad data. We feel that this fact illustrates how difficult it is for an ad network in RightMedia to verify that even local publishers are not engaging in fraud. If the ad exchange suppresses the referrer field, then there is significant lack of accountability in terms of where a publisher can place their section IDs. This loophole provides ample leverage for

a variety of different attacks including misrepresentation using referrer spoofing (see Section 4.2) and clickbots that normally use proxies to obfuscate the source of fraudulent impressions (see Section 2.6).

4.4 Malicious Publishers

By analyzing cookie replay attacks and unknown referrers, we were able to identify a particularly malicious publisher who was the source of a large amount of fraudulent traffic for NetworkX. This publisher, which we call PublisherA, had three section IDs that had already been flagged by RightMedia as generating fraudulent traffic, but were still perpetrating fraud with one section ID that had not been flagged. We first investigated this publisher because it had by far the most unknown referring domains and impressions from these domains, as shown in Tables 4.3a and 4.3b. Because many of the impressions were coming from seemingly random sites, this would indicate a referrer spoofing attack. In addition, we computed that this publisher was generating 20% of the suspicious cookie traffic but accounted for only 0.2% percent of all of NetworkX's local publisher traffic. After being notified of our results, those in charge at NetworkX decided to take action and ban PublisherA from thier network.

PublisherA's historical data gives us an insight into the amount of money a fraudulent publisher can make through a single ad network. PublisherA was part of NetworkX from July 2010 to May 2011, and over that period earned approximately $6,700 on 277,043,885 impressions. This means their eCPM was only $0.02. The fact that they had such a low eCPM is evidence of RightMedia's performance-based pricing, which causes a publisher's CPM to drop if their impressions do not bring measurable revenue to advertisers.

4.5 Fake Web Sites

We observed a number of more sophisticated fraud operations that generated fake web sites with seemingly legitimate domain names. The miscreants then register these sites with different ad networks to become publishers. At first glance, these web sites appeared to contain useful content for web visitors. However, upon further (manual) inspection, we found that there are common patterns to many of these sites. For instance, all of them are based on the same HTML template and, most importantly, contain content that is stolen from other web sites. In addition, the sites appear to be hastily set up with parts of the templates displaying default text (e.g., "text goes here"). Furthermore, we analyzed the WHOIS information for these web site domain names and found very similar registration information (name, phone number, address) across many of these domains.

The botnet C&C server that controlled the malware that we described in Section 2.8 created a number of fake web sites in order to register as a publisher with several ad networks. After receiving section IDs, the malware spoofed the HTTP referrer and directly loaded the HTML iframe that contained the ads that would normally be found on one of these fake web sites. In other words, the malware did not need to visit the fake web sites to load advertisements, but rather could bypass the fake web sites to reduce the amount of bandwidth and hosting costs.

Alexa Page Rank We compared the number of requests per site with the corresponding ranking in Alexa's lists [16]. We chose to use Alexa to assess whether the results we got from the data feed are realistic. Based on our expectations, a web site with higher rank will have a larger number of requests. In most cases, the number of requests and the ranking were similar. However, there were some circumstances where the comparison was disproportionate. In these conditions, we noticed a sharp increase in the number of requests, which were not justified by

Alexa's ranking. These requests constituted approximately 2% of NetworkX's overall traffic.

Chapter 5

Countermeasures

In this Chapter, we propose several countermeasures that can be implemented by ad exchanges to combat the threat of various forms of fraud that we observed.

5.1 Digital Signatures

The main problem that plagues many ad exchanges is the fact that there is an implicit trust in the information that is sent by the browser. In particular, the values sent in the HTTP header fields are trivial for an attacker to manipulate. Thus, ad networks cannot rely on these values without a mechanism to verify the authenticity and integrity of traffic. The primary way to counter this threat is to require ad networks to digitally sign requests, so that forged values can be easily identified. The main drawback of requiring ad networks to sign requests is the additional computation and bandwidth required to encrypt every request.

5.2 Consistency vs. Flexibility

As we discussed earlier, RightMedia does not verify or enforce the basic premise that the referrer must match the publisher's site and assigned sections. The primary

reason for this design appears to be that RightMedia wants their service to be user friendly, and it would be inconvenient if a publisher had to re-register and get new section IDs if they change their domain. We were able to identify one instance of a benign MIA site where the publisher chose to relocate their site to a new domain and keep their old section IDs. Our analysis tools flagged them as suspicious because their original site `benign-golf-site1.com` gave us a 404, but there were a large number of impressions coming from `benign-golf-site2.com`. Manual inspection verified that the site had legal content, and the change of domain most likely came from the owner wanting a more lucrative domain. Third, a number of the most common unknown referrers are legitimate sources of traffic, such as the web site `translate.googleusercontent.com` as well as a number of proxy services. The last reason we suspect RightMedia does not verify the referrer matches the section ID is the attitude that it is up to the ad networks to monitor the validity of their traffic and to filter any potentially fraudulent traffic.

Chapter 6

Related Work

Previous work focused on various aspects of detecting click-fraud. Majumdar *et al.* proposed a content delivery system to verify broker honesty under standard security assumptions [17]. Efficient algorithms for detecting duplicate clicks were proposed by Metwally *et al.* in [18] and Zhang *et al.* in [19]. Studies also have shown how malware can exploit ad networks [6, 20].

Juels *et al.* proposed a cryptographic approach for replacing the pay-per-click model with one where pay-per-action can attract premium rates and unsuccessful clicks are discarded [21]. Immorlica *et al.* studied fraudulent clicks and presented a click-fraud resistant method for learning the click through rate of advertisements [22]. In contrast, Kintana *et al.* created a system designed to penetrate click-fraud filters in order to discover detection vulnerabilities [23].

Recent work has examined botnets and researchers have infiltrated or seized control of parts of the botnet infrastructure to gain more insight into their inner-workings [24, 25, 26, 27]. Note that these botnets were targeted at sending spam email and engaging in acts of financial theft.

In contrast to previous work, our analysis is the first that uses near real-time data to investigate the problem of ad fraud from inside an ad exchange and from the

vantage point of a botnet controller. This offers us a unique opportunity to study the ad exchange structure in depth and to discover its weaknesses. Unfortunately, many ad networks are still reluctant to provide researchers with access to their data streams. As a result, the effectiveness in preventing fraud and even determining the amount of fraud that occurs in actual ad exchanges has not been clear.

Chapter 7

Conclusions

In this thesis, we described how online ad exchanges work and focused in particular on Yahoo's RightMedia. We found that the complexity of the ad exchange provides criminals with an opportunity to generate revenue by developing malware that impersonates legitimate user activities. Regrettably, there is a trade-off between the security of the exchange and the flexibility offered to publishers and ad networks to maximize conversions.

Bibliography

[1] C. Borgs, J. Chayes, O. Etesami, N. Immorlica, K. Jain, and M. Mahdian, "Dynamics of Bid Optimization in Online Advertisement Auctions," in *Proceedings of the International Conference on World Wide Web*, 2007.

[2] A. Metwally, D. Agrawal, and A. E. Abbadi, "DETECTIVES: DETEcting Coalition hiT Inflation attacks in adVertising nEtworks Streams," in *Proceedings of the International Conference on World Wide Web*, 2007.

[3] N. Daswani, C. Mysen, V. Rao, S. Weis, and S. G. K. Gharachorloo, "Online Advertising Fraud," in *Proceedings of Crimeware*, 2008.

[4] B. Edelman, "Securing Online Advertising: Rustlers and Sheriffs in the New Wild West," in *Harvard Business School NOM Working Paper No. 09-039*, 2008.

[5] L. Rodriguez, http://www.washingtonpost.com/wp-srv/technology/documents/yahoo_may2006.pdf, 2006.

[6] N. Daswani and M. Stoppelman, "The Anatomy of Clickbot.A," in *Proceedings of the USENIX Workshop on Hot Topics in Understanding Botnet*, 2007.

[7] M. Gandhi, M. Jakobsson, and J. Ratkiewicz, "Badvertisements: Stealthy Click-Fraud with Unwitting Accessories," in *Journal of Digital Forensic Practice*, 2011.

[8] N. Kshetri, "The Economics of Click Fraud," *Security Privacy, IEEE*, vol. 8, no. 3, pp. 45 –53, May-June 2010.

[9] H. Haddadi, "Fighting Online Click-fraud Using Bluff Ads," *SIGCOMM Computer Communication Review*, vol. 40, April 2010.

[10] "Secure Accounting and Auditing on the Web," vol. 30, no. 1-7, 1998, pp. 541 – 550.

[11] Y. Hu, "Performance-based pricing models in online advertising," *papersssrn-com*, no. March, 2004.

[12] International Secure Systems Lab, "Anubis: Analyzing Unknown Binaries," http://anubis.iseclab.org, 2010.

[13] B. Stone-Gross, A.Moser, C. Kruegel, E. Kirda, and K. Almeroth, "FIRE: FInding Rogue nEtworks," in *Annual Computer Security Applications Conference*, 2009.

[14] B. Stone-Gross, R. Abman, R. Kemmerer, C. Kruegel, D. Steigerwald, and G. Vigna, "The Underground Economy of Fake Antivirus Software," in *Proceedings of the Workshop on Economics of Information Security*, 2011.

[15] "SeleniumHQ. Web Application Testing System," http://seleniumhq.org/.

[16] "Alexa. The Web Information Company," http://www.alexa.com/.

[17] S. Majumdar, D. Kulkarni, and C. Ravishankar, "Addressing Click Fraud in Content Delivery Systems," in *Proceedings of the IEEE Conference on Computer Communications*, 2007.

[18] A. Metwally, D. Agrawal, and A. Abbadi, "Duplicate Detection in Click Streams," in *Proceedings of the International Conference on World Wide Web*, 2005.

[19] L. Zhang and Y. Guan, "Detecting Click Fraud in Pay-Per-Click Streams of Online Advertising Networks," in *Proceedings of the IEEE Conference on Distributed Computing Systems*, 2008.

[20] F. Hacquebor, "Making a Million: Criminal Gangs, the Rogue Traffic Broker, and Stolen Clicks," http://blog.trendmicro.com/making-a-million%E2%80%94criminal-gangs-the-rogue-traffic-broker-and-stolen-clicks/, 2010.

[21] A. Juels, S. Stamm, and M. Jakobsson, "Combatting Click Fraud via Premium Clicks," in *Proceedings of the USENIX Security Symposium*, 2007.

[22] N. Immorlica, K. Jain, M. Mahdian, and K. Talwar, "Click Fraud Resistant Methods for Learning Click-Through Rates," *Internet and Network Economics*, pp. 34–45, 2005.

[23] C. Kintana, D. Turner, J. Pan, A. Metwally, N. Daswani, E. Chin, and A. Bortz, "The goals and challenges of click fraud penetration testing systems," in *In Proceedings of the International Symposium on Software Reliability Engineering*, 2009.

[24] C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. Voelker, V. Paxson, and S. Savage, "Spamalytics: An Empirical Analysis of Spam Marketing Conversion," in *Proceedings of the ACM Conference on Computer and Communications Security*, 2008.

[25] B. Stock, J. Gobel, M. Engelberth, F. Freiling, and T. Holz, "Walowdac – Analysis of a Peer-to-Peer Botnet," in *Proceedings of European Conference on Computer Network Defense*, 2009.

[26] B. Stone-Gros, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna, "Your Botnet is My Botnet: Analysis of a

Botnet Takeover," in *Proceedings of the ACM Conference on Computer and Communications Security*, 2009.

[27] B. Stone-Gross, T. Holz, G. Stringhini, and G. Vigna, "The Underground Economy of Spam: A Botmaster's Perspective of Coordinating Large-Scale Spam Campaigns," in *Proceedings of the USENIX Workshop on Large-Scale Exploits and Emergent Threats*, 2011.