Department of TELEMATIC ENGINEERING
www.it.uc3m.es

Universidad Carlos III de Madrid

Universidad Carlos III de Madrid

Master on Telematics Engineering

Academic Course 2015-2016

Master Thesis

An entropy-based methodology for detecting Online Advertising Fraud at scale

Antonio Ángel Pastor Valles

Directors

Arturo Azcorra Saloña

Rubén Cuevas Rumín

Leganés, September 19, 2016

Keywords: Online Advertising, ad-fraud, Outlier Detection, Information Theory

Summary: Programmatic online advertising has notable benefits for advertisers, but it is highly exposed to fraud. In this thesis, we propose a an efficient and scalable solution to common types of ad-fraud based on the concept of entropy from information theory. For that, we develop a normalized entropic score and describe a lightweight modular system for fraud mitigation that allows not only to filter out evident fraud, but also configure different levels of suspicious activity. The proposed system is configurable letting the advertisers to configure the level of risk they are willing to take. As reducing the risk involves increasing the number of false positives, we propose a multi-level scheme with different price limits to soften the trade-off.

Universidad Carlos III de Madrid. Telematic Engineer Department
Avenida de la Universidad, 30 – 28911 – Leganés (Madrid) – SPAIN
Tel: +34 91 6248778 • Fax: +34 91 6248749
www.it.uc3m.es

An entropy-based methodology for detecting Online Advertising Fraud at scale

Antonio Pastor
Universidad Carlos III de Madrid
anpastor@it.uc3m.es

Rubén Cuevas
Universidad Carlos III de Madrid
rcuevas@it.uc3m.es

Arturo Azcorra
IMDEA Networks Institute
Universidad Carlos III de Madrid
azcorra@it.uc3m.es

ABSTRACT

Programmatic online advertising allows advertisers to diversify their budget dynamically, set the desired context of publishers content, and target a specific audience. Besides, it is accessible to all budgets. Despite these notable benefits, programmatic advertising has the drawback of being highly exposed to fraud.

The detection of fraudulent activities is a difficult task due to the limited information exchanged between ad-networks and the large volume of publishers and traffic sources (IPs). In general, identifying participants in ad-fraud requires a large effort, while recreating a fraudulent system from different IPs targeting new publishers is relatively easy.

In this paper, we propose an efficient and scalable solution to deterministic ad-fraud. The traffic patterns of ad-fraud bots can be identified using the concept of entropy from information theory. We develop a normalized entropic score to identify the domains involved in ad-fraud and the IPs from which the ad-fraud bots operate. We also describe a lightweight and scalable modular system for fraud mitigation that allows not only to filter out evident fraud, but also configure different levels of suspicious activity.

Given the complexity of evaluating the potential fraud, the system is configurable letting advertisers to decide the level of risk they are willing to take. As reducing the risk involves increasing the number of false positives, we propose a multi-level scheme with a bank of bloom filters with different price limits to soften the trade-off.

1. INTRODUCTION

Online advertising drives Internet's economy. A recent report estimates that, only in the European Union, advertising generates the half of online video revenues and the 75 % of the revenue in online journalistic content [22]. In the mobile application market, advertising is the principal funding source and the tendency is to increase its market share. Besides, online advertising revenue has registered double-digit growth in the last six years, accounting nearly 60 billion dollars of annual revenues in 2015 [16].

Online advertising can be divided in traditional and programmatic advertising. The traditional online advertising is reserved to big advertisers and publishers (e.g. websites, apps, online social networks) that can afford the expenses of an agency. Whereas programmatic advertising reduces the entry barriers for small advertisers and publishers. Moreover, in the programmatic model the advertisers can diversify their budget dynamically, set the desired context of publishers content (rather than arranging the campaign with a set of fixed publishers in advance), and target a specific audience.

Advertisers subscribe the services of ad-networks to be generally connected to publishers and vice versa. Similarly, ad-networks connect with other ad-networks through ad exchanges to interconnect their clients to a bigger domain [21].

Despite these notable benefits, programmatic advertising has the drawback of being highly exposed to fraud. The events measured for the common pricing models of programmatic advertising, ad views/impressions and ad clicks, can be easily faked with programs. The detection of fraudulent activities is a difficult task due to the limited information exchanged between ad-networks and the large volume of publishers and traffic sources (IPs). In general, identifying participants in ad-fraud requires a large effort, while recreating a fraudulent system from different IPs targeting new publishers is relatively easy.

In this paper, we propose a lightweight and scalable methodology to identify suspicious traffic patterns applying the concept of entropy from information-theory [17]. The entropy measures the randomness in the activity, and can be applied to identify both publishers and IPs that are more likely to be involved in fraudulent activities.

For instance, suppose a bogus website with poor or duplicated content created with the intention of committing fraud and earning money from the revenues of the contained advertisements. This website will can not generate real traffic by itself and, in order to generate revenue, the creators of the website will generate a large amount of visits with bots loading repeatedly the site from IPs they control. This website will have lower entropy than a legitimate website that receives its visits randomly, from real users with a higher number of different IPs.

The rest of the paper is organized as follows. In the next section we review the related work in ad-fraud research. We introduce the dataset used to develop and validate the methodology in Section 3. Section 4 describes the methodology in detail. Then, in Section 5 we show some validation results and estimations of the fraud identifiable with our methodology. Finally, Section 6 summarizes the main contributions and findings, to conclude with directions for future work.

2. RELATED WORK

Fraud in online advertising can be a highly lucrative business. Data analysis and pattern recognition is necessary for click fraud mitigation and the industry has been applying these techniques for years [10]. As a consequence, the

methods used by ad-fraud cyber-criminals, commonly called fraudsters, have evolve in sophistication as the industry and the research community have been identifying attacks and developing mitigation techniques.

The most simple and straightforward technique used by fraudsters is the creation of a simple website with several ad-frames and then, generate visits to their own website through bots installed on cloud or hosting providers. In a recent paper [5], Callejo et al. analyze how this type of fraud is present on different campaigns they run in Google AdWords. The authors identify up to a 10 % of this type of fraud, depending the targeted keywords.

Other attacks that generate large scale ad-fraud are binded to traffic exchange and reselling. In [19], Springborn and Barford analyze Pay-Per-View networks, exposing how they generate unique visitors to the websites of the traffic buyers rendering these websites underneath the really requested content. Stitelman et al. use co-visitation networks for detecting this type of ad-fraud in [20].

As cost per click (CPC) advertisements usually report higher profits than cost per mille (CPM) impressions, fraudsters simulate click events in websites [14] and mobile apps [7]. The authors of these papers propose techniques for click fraud mitigation serving a percentage of transparent or unattractive ads and analyzing the click through rate to identify the publishers committing artificial clicks.

Another technique for identifying publishers and IPs involved in ad-fraud is an entropic analysis of traffic patterns. The high volume of data exchanged in ad-networks require of lightweight and scalable solutions for data analysis. In this paper, we demonstrate why the entropy fits those requirements and propose a suitable implementation using stream analysis techniques.

The entropy has already been satisfactorily applied by Chen et al. to detect fake views in online video services in [6]. They propose to use the entropy as the final metric and semi-supervised classification with a set of manually labeled samples. A caveat of this approach is that the entropy range is not constant –it varies from 0 to $log_2(k)$, where k is the number of elements in the set. Our methodology extend this basic analysis with a normalized entropic score with an static range from 0 to 100. In addition, we propose an exploratory data analysis method to detect statistical outliers without prior manual labeling and configure different levels of confidence instead using a binary classification.

3. THE DATASET

The dataset we used to develope and validate our methodology has been provided to us by the non-profit ad-fraud research organization Botlab [2]. The data consist in a set of logs from an ad-network, that prefers to remain anonymous, with real traces of ad-requests.

The dataset contains data from 9 days, a sample day with 390 million of entries and two sets of 4 days within a month, the 4 days of each month are evenly separated by a week. The sets covering a month have different fields and number of entries per day, summarized in Table 1. The set with more fields have from 520 M to 560 M entries per day while the other set, that have less fields, have more entries per day, ranging from 1.7 B to 2.1 B.

The distribution of IPs and entries –page loads, removing duplicate entries that could be generated by different ads on the same page– per country is consistent in all the days. The

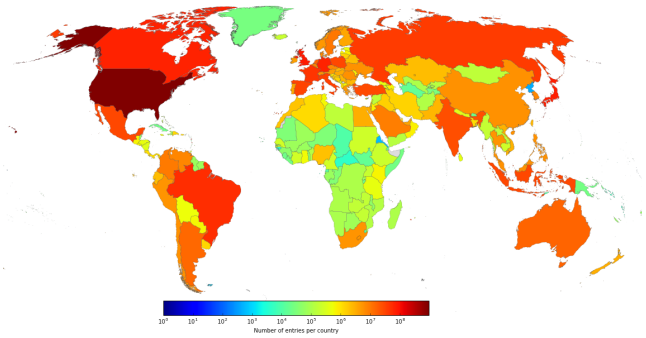


Figure 1: Number of entries (page loads) per country in December 02

United States represents around the 37 % of IPs, followed by the United Kingdom and Japan both with the 5 % of the IPs. If we look to the number of entries, the percentage corresponding to the United States rise to the 47 % while the percentages of the United Kingdom and Japan drop slightly to the 4 %. Figure 1 shows the number of entries per country for the last day of the month set of ∼2 B of entries. We used the MaxMind GeoIP database [13] to identify the countries of the IPs.

3.1 Fields description

Although the name of the fields may be self-descriptive, we provide a brief description of the fields present in the dataset. The fields provide information either of the user or the publisher. The time-stamp, which does not fall within any of the two categories, includes date and time of the entry with seconds resolution. This field is not missing on any entry and does not include time zone. Therefore, we consider that the field is automatically generated on the ad-network upon the reception of the ad requests.

The fields related with the user are the following:

IP The IP address of the user that originates the ad-request. Nearly all the entries have IPv4 values, only 30 entries have IPv6 addresses within all the datasets. However, the percentage of entries with null IP is 8 %. We consider that a domain receiving a high percentage of entries where the IP is null should be checked as an anomaly. Therefore, we include all the entries in the analysis.

UUID The universally unique identifier (UUID) is formed by a 16 bytes number expressed with 32 hexadecimal bits. UUIDs are used as cookie data to identify the user for behavioral advertising. Due to the easiness for users to delete the cookie or to modify the UUID value by a fraudster we do not consider the UUID in our analysis. However, the UUID is useful to identify duplicate entries in the dataset due to multiple ads on the same page. We use the UUID for that purpose when it is present on the dataset.

User-agent A string that identify the browser or app. This field can be used to detect different users or devices using the same IP address, although it can be easily spoofed by fraudsters. It can be used also, like the UUID, to detect duplicate entries on the log.

Table 1: Dataset details: number of entries per day and present fields

dataset	day	# of entries	# of unique entries	Fields									
				time-stamp	IP	UUID	user-agent	inventory type	content type	IAB tier 1 cat	App Name	SSP	Referrer
sample	July 05, 2015	390 M	390 M	x	x	x	x	x	x			x	x
small	September 30, 2015	520 M	512 M	x	x	x	x	x	x	x	x	x	x
	October 07, 2015	558 M	551 M	x	x	x	x	x	x	x	x	x	x
	October 14, 2015	553 M	536 M	x	x	x	x	x	x	x	x	x	x
	October 21, 2015	560 M	551 M	x	x	x	x	x	x	x	x	x	x
big	November 12, 2015	1.94 B	1.68 B	x	x							x	x
	November 19, 2015	2.11 B	1.80 B	x	x							x	x
	November 26, 2015	1.70 B	1.57 B	x	x							x	x
	December 02, 2015	2.14 B	1.94 B	x	x							x	x

The following fields provide information related to the publisher:

IAB tier 1 cat The Internet Advertising Bureau (IAB) defines a hierarchical taxonomy for contextual targeting. The field is a string with the codes of categories associated to the publisher [11]. For the days including this field, it is null in the 26 % of the entries.

App Name For ad requests coming from an app this field indicates the app name.

SSP When the ad-request comes from a supply-side platform (SSP), also called sell-side platform, this field contains its domain. Publishers can use their own SSP software, instead an ad-network, to connect to ad exchanges [4]. Overall, this field match the referrer domain in the 46 % of the entries.

Referrer The domain of the ad-request's referrer. Usually it is the publisher url or the app ID –an string of the form com.Company.ProductName–. The referrer may indicate an ad-network that has pre-bought the advertisement request. The referrer value is missing in the 15 % of the entries.

Finally, the inventory type and content type provide information about the type of ad:

Inventory type The inventory type specifies whether the ad-request of the entry comes from a *website* or an *application*.

Content type The log differentiate the ad requests by the advertisements format. The categories are *display*, *video*, and *native*. The *display* category includes images and gifs, while *native* means the advertisement is integrated as part of the interface of the publisher with a native look-and-feel.

3.2 Data preprocessing

Botlab provided us the datasets in Comma Separated Values (CSV) files. To facilitate the analysis, we uploaded the log files to a postgreSQL database [15]. We use a relational schema with a main table for the log entries and look-up tables to codify the string fields. We store in the main log table an integer mapping to the correspondent string in its respective look-up table to save disk space and speed-up the computations performed by the database when grouping or filtering the data. For instance, checking the equality of two integers requires in general less computations than comparing the characters of two user-agent strings.

For the fraud detection analysis, we are more interested in page loads than in ad-requests. As a web page can have several ads we need to detect those ad requests on the log table belonging to the same page. The ad-requests for the same page should have the same time-stamp, user info, and publisher info. In order to have the page loads accessible, we created a table with the fields that should match on the ad-requests and a count field with the number of ads of each page load. Page loads of the same referrer can differ in the number of ads because a website can have several types of pages with different formats.

We carefully chosed PostgreSQL as the best database for this analysis because it provides a versatile environment rich in datatypes, having data types for IPv4 and IPv6, built-in functions, enhaced SQL syntax, and plugin facilities; while being constrained in space. The functions provided by postgreSQL allow us to extract basic statistics from the logs like counts or maximum and minimum values.

To compute the entropy efficiently and directly on the database we implemented a plugin aggregate function for postgreSQL in the C programming language. For large sets may be more efficient to use a parallelizable approach in an external program or script from a query returning the aggregate counts from the database.

4. METHODOLOGY

The programmatic advertising ecosystem is highly exposed to fraud. The limited amount of information exchanged by the ad-networks and the high volume of ad-requests traffic hinder the identification of those ad-requests not generated by humans, creating a vulnerability in the system. We propose an efficient and scalable solution to simple, but common, types of ad-fraud.

The intrinsic dynamism of Internet suggests data analysis as a natural approach to identify non-human traffic. Analyzing the data traces of users and publishers, at the ad-exchange level, we can identify different patterns and hence, those related to fraudulent activities. We develop our analysis with the following assumptions about known ad-fraud related activities:

Bogus websites and direct-fraud bots Creating a new website is fairly easy and cheap. Fraudsters create websites with poor or copied content but having a high number of ads with the intention of making profit from the advertisements. To generate such profit, the fraudsters run bots performing visits to their own bogus websites. The system is profitable when the revenues are greater than the expenses. Therefore, due the incapacity of these sites to receive human traffic, the number of visits from the bots have to be relevant. The bots can even perform artificial clicks on ads.

Websites buying traffic Websites may artificially increase their daily visits to improve their image and rankings. This practice is performed incognito, hiring a third-party to generate the visits. Websites having advertisements commit ad-fraud when these visits are performed by bots. Even in the case the visit is performed by a real user redirected to the website, the content may remain hidden to the user [1, 19].

Web scraping A web scraper is a program used to extract content of websites simulating the navigation of a human. Some websites that update their content frequently, like news websites, on-line shops, auction sites, sport bets portals, etc. may be repeatedly targeted by programs with different objectives.

Whether ad-fraud is the direct objective of the bots or it is performed indirectly, the advertisements are loaded and billed to advertisers in the same way than when the visitor is a human being. Our objective is to identify the sources (IPs) from which the bots are acting, and the publishers (websites and apps) receiving a high proportion of bot-generated visits.

4.1 Entropy-based detection

These assumptions on fraud related traffic let us model the expected behavior of the bots we want to identify as deterministic. The bots only visit one or a few websites and often. On the other hand, humans are more likely to have a random behavior, visiting several websites only once or a few times each.

Measuring the randomness in the behavior of the IPs we can identify machine patterns. Similarly, websites receiving a significant amount of traffic from bots will have their visits concentrated on fewer sources, while legit websites, only visited by humans, will receive their visits more randomly from any source. Applying the same concept of randomness we can identify the publishers committing ad-fraud and the sources from where such fraud is being generated.

Mathematically speaking, we can measure the randomness with the concept of entropy, more specifically Shannon's entropy, from information theory [17]. Shannon's entropy is the basic measure of information and equivalent to the level of randomness of a signal. The entropy of a discrete random variable $X = \{x_1, x_2, ..., x_n\}$ is defined as

$$H(X) = -\sum_{i=1}^{n} P(x_i) log_2(P(x_i)) \quad (1)$$

The maximum possible value of entropy, $log_2(n)$, is achieved by variables with an uniform probability mass function, where all its possible values have the same probability. In contrast, the entropy is zero, or minimum, when the probability mass function is a Kronecker delta, meaning that there is no uncertainty in the expected value as only one of them can actually happen.

We estimate the probabilities of the elements in the dataset by maximum likelihood: $P(x_i) = C(x_i)/C(X)$, where $C(x_i)$ is the number of occurrences of the element x_i and $C(X)$ is the total number of occurrences of all the elements in the set. Now, we can rewrite the entropy formula to be computed directly from the counts of occurrences of the elements and the total count.

$$H(X) = log_2(C(X)) - \frac{\sum_{i=1}^{n} C(x_i) log_2(C(x_i))}{C(X)} \quad (2)$$

The resulting formula is easily parallelizable with map-reduce, where the operations inside the additivity can be implemented in a map function with a final reduce function for adding the values resulting from each element map computation and compute the final value. Besides, storing in advance the joint occurrences of IPs and domains from the ad-requests stream, we can design a system to compute the entropy from the occurrences count in linear time. Similarly, we can either compute the total occurrences of each IP or domain at the time we compute their entropy or do it in advance from the data stream.

Note that we can skip the computations in the map step of all the elements with count one, as its logarithm is zero. In practice, on a one day window, having a count of 1 for <IP,domain> tuples is very common. This particularity reduces significantly the computation time. The space needed may be reduced as well using a bloom filter for storing only tuples after the second occurrence.

Although, given the significant amount of possibles IPs, it may be interesting to consider other approaches that could allow to estimate the counts distribution on a constrained storage space, like using approximate frequency counts over data streams [12]. In order to compute the entropy, we are interested in the probabilities distribution rather than in the individual values of those probabilities. Therefore, we can still compute an estimation of the entropy storing only a certain number of tuples per domain and the total number of entries.

4.2 Normalized entropic score

The entropy let us measure the level of randomness on the expected visitor of websites and in the behavior of IPs. However, the upper-bound value for a totally random behavior depends on the number of elements in the set. Therefore, a more standard range of values would be desirable. In this section we show how we developed a normalized score based on the entropy. Although the normalization can be equally applied to IPs and websites entropy, we will use the websites score to illustrate the normalization process.

The entropy of a domain is upper-bounded by the number of IPs that visit such domain. Accordingly, we use the number of total visits received by a website as estimator of the number of potential visitors. The maximum randomness possible in the expected visitor for a website that receives k visits is achieved when each visit is received from a different source. Similarly, the maximum entropy achievable by a website with k visits is $log_2(k)$, when each visit comes from a different source.

Table 2 shows the entropy of three websites computed as described in the previous section. The *domain 2* and the *domain 3* have the same entropy because their probability of receiving a visit from any of the five IP sources in the set is the same. Despite this, the *domain 3* has 250 visits, which may seem a considerable number for being generated from only 5 sources, while the *domain 2* has only 5 visits. This example illustrates that we cannot say if a value of the entropy is good or bad without taking on account the total number of entries that the domain receives.

We obtain a more fair metric normalizing the entropy by

Table 2: Entropy of different websites according to the distribution of their visits

	IP_1	IP_2	IP_3	IP_4	IP_5	Entropy	Score
domain 1	5	0	0	0	0	0	0
domain 2	1	1	1	1	1	2.32	100
domain 3	50	50	50	50	50	2.32	0.29

the total number of entries received by the site. The normalized score corrects the entropy value with the binary logarithm of the total number of entries

$$H(X) = 100\left(1 - \frac{\sum_{i=1}^{n} C(x_i)log_2(C(x_i))}{C(X)log_2(C(X))}\right) \quad (3)$$

Figure 2 illustrate the motivation to use a normalized metric instead of the entropy as explained above. The upper-bound of the range of possible values for the entropy is determined by the total number of entries that a website has received. Therefore, the area of suspicious activity should be delimited by a line. The suspicious activity threshold line can be computed using robust outliers detection. The domains can be divided in buckets by their total number of entries. Then, the threshold can be configured as the linear regression of the lower caps values of the boxplot of each bucket, like shown in fig. 2a. Similarly, the threshold could be configured in ladder using the lower cap value of each bucket, but this technique is less robust and could lead to use an incorrect value on a bucket with few data. Figure 2b shows how the normalization flattens the distribution over the buckets allowing to use a single value threshold to define suspicious activity of the domains.

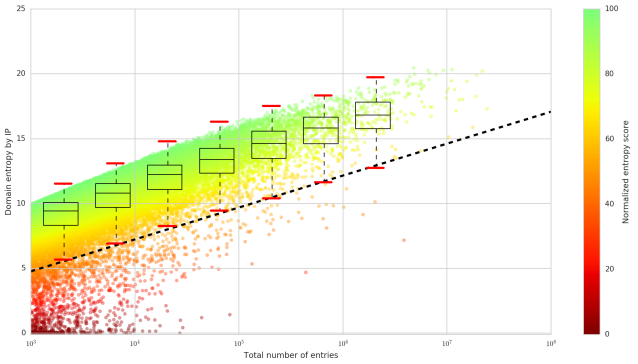
4.3 Threshold selection

Once we have a metric which distribution is independent of the number of entries, the next question is where to set a fair threshold to separate suspicious activity from the legitimate human traffic. We propose an exploratory data analysis approach for robust outliers detection based on the boxplot method and propose different thresholds for different degrees of suspicion.

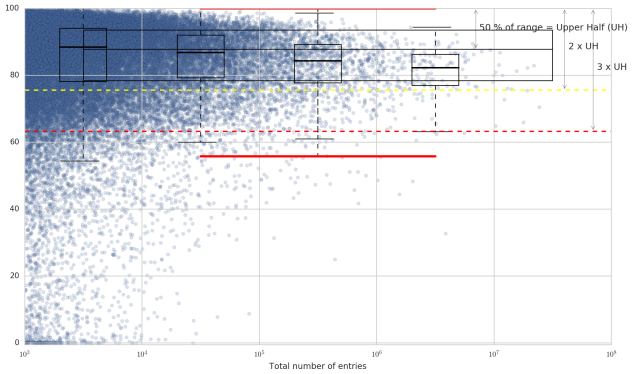
The boxplot is widely used as a robust method for outliers detection because it is not affected by extreme outliers, like the standard deviation, and several modifications have been proposed in the literature for non-symmetric distributions [9, 18]. The boxplot uses the interquartile range as a measure of the dispersion of the distribution. A measure of dispersion that is valid while the outliers on either sides of the median does not exceed the 25 %.

In our context, it is not safe to make such assumption for low scores, so we need a another measure for the dispersion of the distribution. In contrast, a safe assumption for the domains score is that having a high score is not suspicious per se. It should be common that websites only receive one entry a day of each user and the data validate this hypothesis showing no potential outliers on the upper side of the boxplots at any range. In fig. 2 the upper cap is placed at the minimum of either 1.5 IQR or the maximum value present in the data, explaining that some boxplot's caps do not reach the theoretical maximum.

Assuming that the 50 % of the data with higher score is generated from human traffic, we can use as measure of dispersion the upper half (UH) range, the distance between the median and the maximum value, of the score distribution. We set a threshold for slightly suspicious scores at twice the upper half range from the maximum and another threshold for the next level of suspicion at three times the upper half range from the maximum. Finally, we set a third threshold for highly suspicious scores at the lower cap of the traditional boxplot.



(a) Scatter plot of domains' entropy by IP vs. the total number of visits received by the domain. The colorbar shows the corresponding normalized score.



(b) Scatter plot of domains' normalized entropic score by IP vs. the total number of visits received by the domain

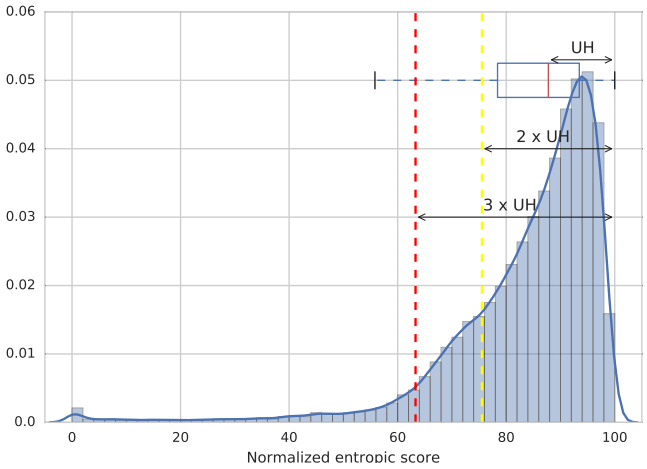
Figure 2: Scatter plots with results of domains with more than 1 thousand visits the December 02. The boxplots show the distribution of the entropy and the score dividing the domains in buckets in function of the number of visits.

Figures 2b and 3a show the threshold values for the domains on December 02. The histogram of the domains (fig. 3a) is highly skewed towards very high values, proving that the majority of the domains are visited randomly from different users. And therefore, supporting our hypothesis for legitimate domains visited by humans. However, the distribution of the IPs score (Figure 3b) is more centered, suggesting that the alternative thresholds suggested above for the domains may not be suitable for the IPs.

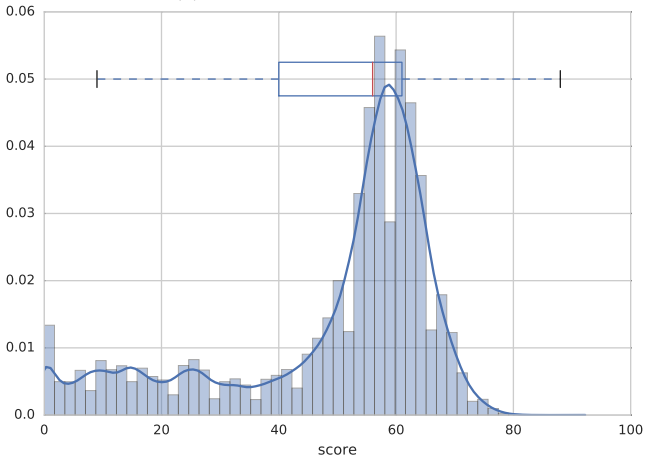
4.4 System design

The ad-fraud detection methodology we describe in this paper can be easily implemented for its use in production. The system can be used by ad-exchanges, ad-networks, or agencies and it is composed of the following three main tasks:

(1) Log the ad-requests received by the system.



(a) Domains score distribution



(b) IPs score distribution

Figure 3: Distribution of the score of the domains and IPs with more than 1 thousand entries the December 02

(2) Analyze the data to compute daily scores.

(3) Update the filters that will categorize the ad-requests by its degree of suspicion.

An overview of how our anti ad-fraud system could be deployed is depicted in Figure 4. The ad-request is triggered by the user browser when the user visits a webpage containing in its html code an url to retrieve an advertisement. The ad-request is forwarded from the user machine to the ad-fraud filter server which will decide if the ad-request is legitimate and should be forwarded to the ad-exchange. The ad-fraud filter server will also forward the ad-request to the data analysis system independently of the filtering decision. At midnight, the data analysis system will switch the data collection to a new database and will start computing the new scores from the ending day data. The data analysis system will send the new scores to the filter server when their calculation is completed.

In this example the system is deployed at the ad-exchange level, but it is equally useful for ad-networks or agencies. Similarly, the window time for computing the new scores can be configured at a different intervals or even use a sliding scheme. For instance, every 8 hours the scores are computed

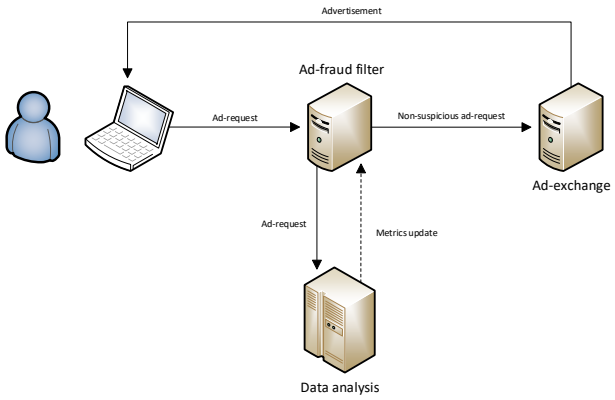


Figure 4: Ad-fraud system example diagram

from the data of the last 24 hours.

An interesting feature of this approach for ad-fraud mitigation is that the filter does not necessarily have to be a binary classifier. We can configure a bank of bloom filters in parallel for different levels of distrust. The ad-exchanges or the agencies can set different maximum prices they are willing to take for each level of the bank filter. Ad-requests classified in last category, for the most suspicious traffic, can be directly discarded, saving the client of receiving junk traffic.

Due to the incapacity to create a perfect ad-fraud detection system, the multi-level classification of the ad-requests let the advertisers decide the risk they are willing to take on their campaigns. The advertisers, or the agencies on their behalf, can set different price limits at each level of suspicion and modify those limits according to their budget tracking throughout the campaign duration.

5. RESULTS

In this section we compute our entropic score for the domains and traffic sources of the Botlab dataset proving that low scores are linked to evident fraudulent activities. We also show the percentage of traffic that would be filtered at each level of suspicion.

Given the huge amount of daily traffic and the dynamic behavior of the fraudulent activity, it is not practical to check manually the results for setting a threshold that would minimize the classification error. We let for future work the study of an automated evaluation method derived from analyzing further characteristics, like the inter-event times distribution.

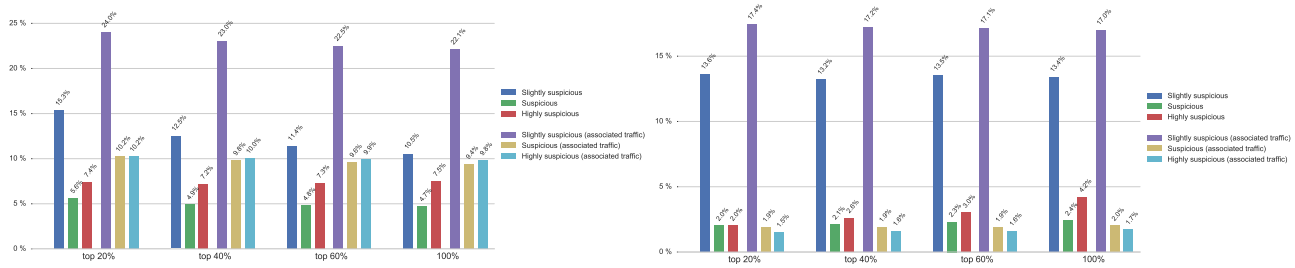
In order to check the stability of the results across time we analyze each day separately. This is necessary to ensure the fraud-detection system is applicable to a production system. We found that both the scores and number of entries per domain or IP distributions are stable. Hence, we will focus on the results of two days: the October 21 for results from days with the user agent field and December 02 for the days with more entries.

5.1 Domain results

Although manually checking the logs is not practical, we show some examples of domains with low scores where we manually identified, with high confidence, that they are committing ad-fraud. Table 3 shows three domains where we

Table 3: Domains identified committing ad-fraud with low entropy

domain	# visits October 07	# visits October 14	# visits October 21	score October 21
automotiveartgallery.com	402	4	7810	21
animeoscar.xyz	394	1079	7159	14
waznemiejsca.blogspot.com	8607	9411	10088	14



(a) October 21, 2015

(b) December 02, 2015

Figure 5: Potential fraud in websites with more than 1 thousand visits. The subgroups of the x-axis are formed according to the number of visits received by the websites.

confirmed that the visits were being performed only from a few IPs throughout the day. Casually, if an IP stops visiting the website at a random time another IP in the same subnetwork starts visiting the website around 5 minutes later.

The scheme used by *animeoscar.xyz* and *automotiveartgallery.com* is the same. Both sites receive the visits from IPs in the subnetwork 209.95.50.0/24, registered to a hosting provider in the United States, and the bots seem to vary randomly the user-agent, for instance, one IP that performs 2.384 entries has 1.960 distinct user-agents. The blog *waznemiejsca.blogspot.com* uses simple bots from 2 IP belonging to 2 different Polish Internet Service Providers (ISPs) with 3 and 2 user agents respectively.

In general, mobile apps have lower scores than websites. This could be explained because there are some apps, like games, where the user spend more time and therefore will see more adds from the same publisher. However, there are some IPs with a number of entries that would require various users connected all day to the app through the same IP. In this case, an analysis of the IP activity could help to unravel bots from real users on the same private network. The fraud in mobile advertising could be committed either running apps continuously on real phones or emulating the app in a computer.

There are platforms that reward their users for the viewed advertisements. Some users of these platforms develop hacks to increase the rewards automatically without having to see the adds. For instance, the app *com.perk.screen*, that rewards its users for the adds seen on the unlock screen, has 1 million entries the October 21 and a score of 14. This app has showed nearly 45 thousand ads to one IP, meaning that has served 31 ads per minute to that IP. There are 32 IPs with more than 10 thousand entries within a day –10 thousand entries correspond to an average of 7 entries per minute.

We also found on October 21 the following two websites with a different pattern of suspicious behavior: *hawaiidiveresorts.com* and *mystarhomes.com*. They received around 500.000 visits each one with scores of 62 and 61 respectively. Both websites does not exist anymore and on the datasets of previous days have a number of entries much lower. Both websites receive their visits from several IPs performing the majority of them a significant number of entries in a few hours, all of them using Google Chrome or Safari as browser. The visits pattern suggests that the fraudsters were using infected PCs or browsers and we estimate that the bots were configured to generate around 8 visits per minute.

Altogether, Figure 5 shows the percentage of potential fraudulent traffic that would be filtered using our entropic score and proposed thresholds. Note that these numbers vary with the fraudulent activity of the types that our system detects, described in Section 4. Given the dataset in December 02 has four times the number of entries than in October 21 and that should contribute to increase the accuracy, we consider the results of December 02 a better estimation.

5.2 IP results

A common evaluation metric for ad-fraud is that a significant amount of fraud sources belong to datacenters[1]. Unfortunately, the public repositories with IP prefixes of datacenters include a limited subset of them. We use two public repositories [3, 8] to tag the IPs present on the dataset that belong to hosting providers. Despite only the 1.5 % of the December 02 entries belong to datacenters (4.3 % of the entries for the domains with more than 1 thousand visits), we find that the 84 % of the entries from datacenters are to domains with the score below the highly suspicious threshold and the 99 % to domains bellow the slightly suspicious threshold. The number of visits from datacenter IPs to domains with more than 1 thousand visits the December 02 is 14 million.

Figure 6 show separate boxplots for the score of the IPs according they belong to a datacenter or not. Note that not all traffic generated from datacenters may follow the patterns of navigation described in Section 4. For instance,

[1]Note that for simplicity we use the term datacenter to refer to both Cloud and Hosting providers.

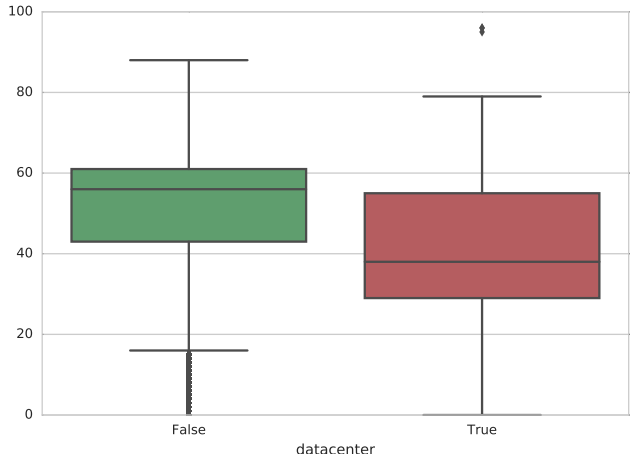


Figure 6: Boxplots of score of the IPs with more than 1 thousand entries the December 02. The IPs are divided according they belong to a datacenter or not.

the outliers that the red boxplot mark with a high score are 3 IPs crawling multiple blogs from *wordpress.com* –the urls of Wordpress blogs are formed using subdomains and we do distinguish subdomains for computing the scores. However, it can be appreciated a difference in both distributions.

Adding to the datacenter IPs group those of the other group which are outliers with a low score, the percentage of entries that could be flagged for potential ad-fraud are the 14.8 % and their associated traffic is the 13.5 %. These results are for December 02 dataset for IPs that generate more than 1 thousand entries.

6. CONCLUSIONS

Programmatic advertising has intrinsic vulnerabilities. The dynamic nature of Internet requires data driven solutions for fixing those vulnerabilities. While the techniques used by fraudsters have evolve to avoid blocking and generate more profit, new attackers can take advantage in the future of the basic vulnerabilities if the ad-networks are not properly protected. The dynamic nature of Internet makes data analysis a good approach to mitigate ad-fraud and the high volume of traffic exchanged require scalable solutions.

We propose a modular and scalable system for fraud mitigation. The system uses a lightweight and parallelizable entropic metric to detect deterministic ad-fraud. The proposed solution is compatible with additional techniques for the detection of fraudulent activities.

Although it is not possible to provide an accurate evaluation of the system's accuracy, the system is configurable letting advertisers to decide the level of risk that they are willing to take. As reducing the risk involves increasing the number of false positives, we propose a multi-level scheme with different price limits to soften the trade-off.

The results show that the percentage of evident suspicious activity detectable measuring the level of randomness of publishers visitors is limited. However, analyzing the traffic patterns of IPs and blocking those IPs from hosting providers, the filtered fraud rise over the 10 %. The slightly suspicious traffic is considerable and further analysis of the time between visits could help to identify the real fraud in the range of slightly suspicious score values.

We consider these numbers a pessimistic estimation of the fraud present in the programmatic advertising ecosystem due to the limitations of the methodology. Our intuition is that fraudsters are masking their activities more carefully, either using more IPs at a lower frequency or more domains with a lower number of visits to them. Therefore, we propose as further research directions the development of techniques for the detection of low frequency fraud.

References

[1] Unmask Parasites Blog. *Evolution of Hidden Iframes*. URL: http://blog.unmaskparasites.com/2009/10/28/evolution-of-hidden-iframes/ (visited on 08/15/2016).

[2] *Botlab*. URL: http://botlab.io (visited on 08/01/2016).

[3] Botlab. *Deny Hosting IP repository*. URL: https://github.com/botlabio/deny-hosting-IP (visited on 09/01/2016).

[4] Internet Advertising Bureau. *Advertising Ecosystem*. URL: http://www.iab.net/data/ecosystem.html (visited on 09/01/2016).

[5] Patricia Callejo et al. "Independent Auditing of Online Display Advertising Campaigns". 2016.

[6] Liang Chen, Yipeng Zhou, and Dah Ming Chiu. "Fake view analytics in online video services". In: *Proceedings of Network and Operating System Support on Digital Audio and Video Workshop*. ACM. 2014.

[7] Geumhwan Cho et al. "Combating online fraud attacks in mobile-based advertising". In: *EURASIP Journal on Information Security* (2016).

[8] Nick Galbreath. *ipcat: datasets for categorizing IP addresses*. URL: https://github.com/client9/ipcat (visited on 09/01/2016).

[9] Mia Hubert and Ellen Vandervieren. "An adjusted boxplot for skewed distributions". In: *Computational statistics & data analysis* (2008).

[10] Brendan Kitts et al. "Click fraud detection: adversarial pattern recognition over 5 years at Microsoft". In: *Real World Data Mining Applications*. Springer, 2015.

[11] IAB Technology Lab. *OpenRTB API Specification Version 2.4*. 1, 2016.

[12] Gurmeet Singh Manku and Rajeev Motwani. "Approximate frequency counts over data streams". In: *Proceedings of the 28th international conference on Very Large Data Bases*. VLDB Endowment. 2002.

[13] MAXMIND. *GeoLite2 Free Downloadable Databases*. URL: http://dev.maxmind.com/geoip/geoip2/geolite2/ (visited on 08/08/2016).

[14] Brad Miller et al. "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*. Berlin, Heidelberg: Springer-Verlag, 2011.

[15] *PostgreSQL*. URL: http://www.postgresql.org (visited on 08/11/2016).

[16] PricewaterhouseCoopers. *IAB internet advertising revenue report. 2015 full year results*.

[17] C. E. Shannon. "A mathematical theory of communication". In: *The Bell System Technical Journal* (1948).

[18] Georgy Shevlyakov et al. “Robust versions of the Tukey boxplot with their application to detection of outliers”. In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE. 2013.

[19] Kevin Springborn and Paul Barford. “Impression Fraud in On-line Advertising via Pay-Per-View Networks”. In: *22nd USENIX Security Symposium (USENIX Security 13)*. USENIX, 2013.

[20] Ori Stitelman et al. “Using co-visitation networks for detecting large scale online display advertising exchange fraud”. In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2013.

[21] Brett Stone-Gross et al. “Understanding Fraudulent Activities in Online Ad Exchanges”. In: *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*. ACM, 2011.

[22] IHS Technology. *Paving the way: how online advertising enables the digital economy of the future*. 25, 2015. URL: http://www.iabeurope.eu/wp-content/uploads/2016/01/IAB_IHS_Euro_Ad_Macro_FINALpdf.pdf (visited on 07/26/2016).