# Annoyed Users: Ads and Ad-Block Usage in the Wild

Enric Pujol
TU Berlin
enric@inet.tu-berlin.de

Oliver Hohlfeld
RWTH Aachen
oliver@comsys.rwth-aachen.de

Anja Feldmann
TU Berlin
anja@inet.tu-berlin.de

## ABSTRACT

Content and services which are offered for free on the Internet are primarily monetized through online advertisement. This business model relies on the implicit agreement between content providers and users where viewing ads is the price for the "free" content. This status quo is not acceptable to all users, however, as manifested by the rise of ad-blocking plugins which are available for all popular Web browsers. Indeed, ad-blockers have the potential to substantially disrupt the widely established business model of "free" content—currently one of the core elements on which the Web is built.

In this work, we shed light on how users interact with ads. We show how to leverage the functionality of AdBlock Plus, one of the most popular ad-blockers to identify ad traffic from passive network measurements. We complement previous work, which focuses on active measurements, by characterizing ad-traffic in the wild, i.e., as seen in a residential broadband network of a major European ISP. Finally, we assess the prevalence of ad-blockers in this particular network and discuss possible implications for content providers and ISPs.

## Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols; H.5.4 [**Information Interfaces and Presentation**]: Hypertext/Hypermedia

## Keywords

Web; Advertising; AdBlock Plus; Residential Broadband Traffic

## 1. INTRODUCTION

The World Wide Web has fueled an unprecedented commercialization of the Internet by turning a system designed for academic data exchange into a widely used social medium. The economic foundation of many services in this medium is online advertisements (ads) which allow content and services to be offered free of charge to users.

The history of this business model goes back to the first clickable Web ads, which emerged around 1993 with the first commercial Web sites. HotWired was among the first to sell banner ads to companies such as AT&T and Coors. The first central ad servers emerged in 1995 to enable the management, targeting, and tracking of users and online ads. This started the proliferation of Web ads, which brought with it an increasingly complex infrastructure to serve these advertisements. DoubleClick introduced the process of online behavioral advertising in the late 1990s which used 3rd party cookies to track users across sites and present ads based on the users' browsing patterns. Today, online advertisement has become a large and complex industry where entire exchanges trade user-specific information for the purpose of better ad placements [58].

This user specific profiling has raised many privacy and security concerns in particular among privacy and consumer advocacy groups. The debate has, for instance, led to agreements about the expiration dates of *cookies* as well as clear statements regarding the collection of user information [8]. While the Federal Trade Commission (FTC) identifies problems with online behavioral advertisement, it allows advertisers to continue this practice with some safeguards such as greater transparency, provisions for consumers to opt out, and special handling of sensitive data, e.g., those related to health and financial information [13]. The motivation underlying the FTC's position is that the FTC views online advertisements as the key enabler for "free" content on the Web.

However, to Web users, online advertisements can be perceived as being not only invasive to privacy, but also annoying, since they can distract them from the primary content they wish to consume. This situation has resulted in a proliferation of tools to evade or block the ads. We refer to these tools as *ad-blockers*. Among the most convenient and popular ad-blockers are extensions for Web browsers like *Adblock Plus* and *Ghostery*. According to Google and Mozilla usage statistics of browser add-ons, *Adblock Plus* is among the most popular. More than 30M users surf the Web daily using a browser with this extension enabled [18, 15].

The use of ad-blockers is, however, perceived by the advertisement industry and content publishers as a growing threat to their business model [6, 19]. Their rationale is that ad-blockers provide users a way to "evade paying" for the content they consume. Thus, some players try to put pressure on the developers of ad blocking browser extensions in order to be excluded from the blocking. This exclusion is envisioned to be implemented by *i)* removing them from the filter lists, or *ii)* by adding them to a whitelist [6]. Others also try to detect ad-blockers and explicitly appeal these users to either whitelist their site or completely disable the ad-blocker.

Another claim regarding the rise of ad-blockers is that "as more end users adopt them, revenues decline and the number of obnoxious advertisements increases" [3]. To break this vicious cycle

*IMC'15,* October 28–30, 2015, Tokyo, Japan.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3848-6/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2815675.2815705.

some players in this domain encourage advertisers to adopt "acceptable ads" as a compromise. The most important implication of this initiative is that advertisements that conform to these guidelines are by default whitelisted by *Adblock Plus* [5], a setting that end users can still override.

Moreover, the ad-blocking debate is not uniquely centered on adblockers like *Adblock Plus*. In fact, some Internet Service Providers (ISPs) have already stated their intention to block advertising on their networks and thereby, force some companies in the advertising business to share their revenues [12]. In light of this tussle between users, advertisers, publishers, and ISPs, we inform the ongoing debate with a passive measurement study about the "ad-scape" and how end users interact with it.

**Our work:** We complement previous work with a passive measurement study on advertisements. Namely, we collect traces from a residential broadband network (RBN) of a major European ISP to *i)* elaborate on the prevalence of ad-blockers in this vantage point, and *ii)* characterize ad traffic in the wild. The contributions of this paper are as follows:

- We present a methodology to extract structural Web site information from packet header traces. With this methodology at hand, we can use the *Adblock Plus* functionality to classify ad traffic in passive measurements. We validate the effectiveness of this methodology with an active measurement study based on different configurations of an instrumented browser, e.g., running *Adblock Plus*.
- We leverage two indicators to infer *Adblock Plus* usage: *i)* low ratio of ad requests, and *ii)* connections to the *Adblock Plus* servers. Based on these metrics we estimate that 22% of the most active users likely use this extension. Moreover, we have indications that, perhaps, most *Adblock Plus* users do neither subscribe to *EasyPrivacy* (the list that protects them from trackers), nor opt out from the list of *non-intrusive* ads.
- We find that advertisement traffic contributes to a significant percentage of requests, roughly 18% in both of the studied traces. We dissect ad requests by type and find that 11% of them match the list of *non-intrusive ads*, while the rest is distributed among advertisers and trackers. Given the prevalence of this traffic in terms of requests, we also characterize various aspects thereof, including object types, server-side infrastructure as well as back-end functionality, i.e., real-time bidding.

The remainder of this paper is structured as follows: we first discuss the basic functionality of *Adblock Plus* and the corresponding filter lists in §2. We then describe in §3 our measurement methodology and its subsequent evaluation in §4. Section §5 summarizes the two data sets as well as the involved vantage points. We present our first results about *Adblock Plus* usage in §6. We then elaborate on traffic- and infrastructure-centric aspects of advertisements in §7 and §8. Section §9 presents the related work. We discuss the limitations of our methodology in §10 and present our conclusions in §11.

## 2. AD BLOCKERS – A SHORT REVIEW

There is a wide range of browser extensions available to end-users who want to evade or protect themselves from the *ad-scape*. *Adblock Plus* is arguably among the currently most popular adblockers [17]. Users can configure this browser extension to *i)* block or hide advertisements, and to *ii)* protect their privacy by blocking trackers. Another popular tool is *Ghostery*, which mainly focuses on protecting end-users' privacy. The Electronic Frontier Foundation's *Privacy Badger* [10] shares the same objective; but in contrast to *Ghostery*, which is proprietary software, *Privacy Badger* is open source—and based on *Adblock Plus*. Another option is the *NoScript* plugin which is designed to interactively disable executable Web content such as JavaScript, which is often used by advertisers and trackers.

To assess the popularity of these extensions we can refer to statistics reported by popular browsers or to recent work by Metwalley et al. [46]. They report, relying on a passive measurement study, that 80% of the households visible at their vantage point do not use any of these popular plugins. Among those that do, *Adblock Plus* dominates, i.e., *Adblock Plus* is installed at 10%-18% of the households. Indeed, less than 3% of the households exhibit evidence of other installed plugins. Given the popularity of *Adblock Plus*, we next describe how this particular extension operates.

***Adblock Plus*.** At the heart of *Adblock Plus* is the mechanism to filter adverts based on *filter rules* that appear in *filter lists*. The regular expressions that form the set of filter rules follow a specific syntax, which is described in detail in [4, 21]. If a filter rule matches a URL that is not otherwise whitelisted, *Adblock Plus* will prevent the Web browser from requesting the URL. Hence, this extension reduces network traffic as it averts undesired ad-related objects from being fetched.

However, some Web pages embed advertisements into the (main) HTML document, e.g., textual advertisements. Since this document is required to render the page, the tool does not block the download of these HTML documents. However, *Adblock Plus* includes functionality to hide such—otherwise displayed— embedded ads via CSS modification. Note that these ads are still transferred over the network even though they will not be displayed in the browser to the user.

**Filters Lists.** *Adblock Plus* users can obtain various lists of filter rules via a subscription mechanism. There are several filter lists available for different purposes. When an end user installs the *Adblock Plus* extension for the first time, the plugin subscribes itself to two filter lists. The first one is named *EasyList* and its goal is to remove ads from English Web pages. The second list is called *non-intrusive advertisements* (acceptable ads) and its purpose is to whitelist the advertisements that are blacklisted by *EasyList* but comply with the directives summarized in [5].

This list and, in particular, its activation by default is most likely the origin of the controversy described in [6] (i.e., advertisers paying to be whitelisted). Note that users may still opt to deactivate this list with a single click. There are additional lists to which *Adblock Plus* users can subscribe. Examples include *i)* customizations of *EasyList* to non-English Web pages, or *ii) EasyPrivacy*, which aims to protect the end users' privacy by blocking Web trackers.

**Measurement challenges.** At first glance, given that this adblocker only uses filter lists, it should be easy to emulate its behavior on a passively collected network trace. However, this is not that simple as *Adblock Plus* relies on the information contained in the DOM tree of the Web site to classify Web elements as adverts. For example, the classification can be based on whether an image is displayed in an iframe, which cannot be detected by inspecting the URL. Thus, this ad-blocker relies on the entire structure of a Web page rather than purely the URLs.

To reconstruct the entire structure of the Web page one needs parse the payload part of the traces. For privacy reasons we cannot and do not have access to this part. Our traces only include HTTP header information (see §5). Hence, our methodology has to rely on the information available in the HTTP headers. How we tackle this challenge is discussed in the next section. Our methodology enables us to *approximately* reconstruct the Web page meta-data from the HTTP headers.

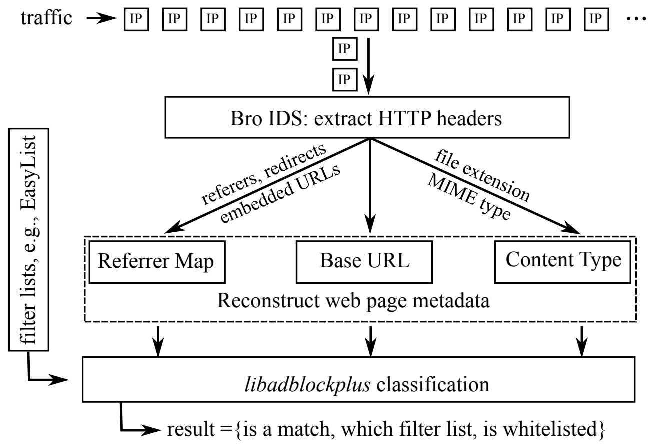

traffic
IP
Bro IDS: extract HTTP headers
referers, redirects
embedded URLs
file extension
MIME type
filter lists, e.g., EasyList
Referrer Map
Base URL
Content Type
Reconstruct web page metadata
libadblockplus classification
result ={is a match, which filter list, is whitelisted}

**Figure 1: Approach to classify ad requests.**

# 3. METHODOLOGY

In this section we describe our approach for identifying ad-blocker users from passive traces. The premise is that ad-blocker users access fewer ads than non ad-blocker users. Thus, we need a methodology for identifying Web objects and separating them into ad or non-ad objects. This is where we rely on *Adblock Plus*.

## 3.1 Identifying ad traffic

The goal of our methodology is to classify ad traffic in TCP/HTTP header traces captured via passive measurements. For this classification we rely on *Adblock Plus* functionality as it is the most popular ad-blocker to date. But rather than using a *Adblock Plus* browser directly we use *libadblockplus* [16], a C++ wrapper around *Adblock Plus*, developed by the same project. This allows us to classify Web objects from the traces in an off-line fashion without the need to operate a full browser. A summary of the classification methodology is shown in Figure 1.

We use *Bro* [48] HTTP analyzer to extract information about all HTTP transactions in network traces. This information includes *i)* both the `Host` and the `URI` fields in the request header, *ii)* the `referer` field in the request, *iii)* the `Content-Type` field in the response, and *iv)* the `Content-Length` field present in HTTP responses. We also extended the *Bro* analyzer to parse and include *v)* the `Location` header field present in response headers that relate to HTTP redirections.

Processing passive traces with *Bro* gives us a list of Web objects. We then invoke *libadblockplus* to classify each of these objects into ad and non-ad objects. However, *libadblockplus* cannot properly classify a URL as an advert using only the information that is self-contained in the URL's string. Many rules in the filter lists apply to specific combinations of domains, i.e., a Web object hosted in a specific (advertiser) domain from a specific (publisher) domain. Thus, *libadblockplus* requires the following information to properly classify a URL: *i)* the requested URL itself, *ii)* the rest of URLs in the Web page that triggered the request that is currently processed, and *iii)* the type of the content that is being requested e.g., `document`, `script`, `stylesheet`, `image`, `media` or `object`.

As mentioned above, the *Adblock Plus* browser extension relies on information contained in the DOM tree. We, in contrast, have only the information available in the HTTP headers. However, we can still use this information to obtain a *partial view* of the relationships between the Web objects in a Web page. We tackle this challenge in the following way. See the middle boxes in Figure 1 for an illustration of the approach.

**Referrer Map.** First, we extract the set of related URLs for a given request. To this end we construct a *referrer map* that approximates the set of URLs in a Web page based on the chain of observed HTTP referrers. Our approach is based on the *StreamStructure* and *ReSurf* methods discussed in [38, 56]. We also use these methods to obtain the Web site that triggered the request. We construct the referrer map out of the values in the `referer` header fields in the requests. However, there are a few cases in which this chain may be "broken". One typical example is when the request following a redirection to a new URL has no `referer`. This is the reason why we extend *Bro* to also parse the `Location` response headers. With this small modification we can add this type of missing referrers to the map of referrers. Furthermore, we also insert the URLs that we can find embedded within the URL of a request into the referrer map.

**Content Type.** Second, we infer the type of the content in the following way: one of the pitfalls in HTTP traffic analysis are mismatches between the `Content-Type` of the request and the actual content. Schneider et al. [52] showed that while mis-matches often occur due to the format (e.g., `jpeg` vs. `png`), they actually agree on the general category (e.g., `image`). In these cases the mismatch does not impair our classification because *libadblockplus* relies on general categories. For other mismatches we parse the URL to map the following file extensions to content types: i) *.png*, *.gif*, *.jpg*, *.svg*, *.ico* (`image`) ii) *.css* (`stylesheet`) iii) *.js* (`script`) iv) *.mp4* and *.avi* (`media`). As a rule of thumb, we rely on the `Content-Type` field when the file extension does not yield a type. Some redirections may lead to mis-classifications. For instance, there are cases where a URL within an `<img>` HTML tag results in a redirection. Suppose there exists an exception filter for that URL and its content type is `image`. To *Adblock Plus* this request is an image, since it can glean this information from the tag. We, on the other hand, would filter it because we do not have access to this information. Here, the referrer map helps us to set the appropriate content type for the URL that is being redirected by inspecting the type of the consequent request.

**Base URL.** Third, we process the URLs to avoid conflicts with the filter lists. Namely, we noticed that *libadblockplus* mis-classifies some requests because they include parts of the URL of a previous request in the query string. While the *Adblock Plus* plugin does not filter the second request, *libadblockplus* does. To prevent this type of mis-classifications we normalize the query strings by removing dynamic values. However, there are some filters in these lists that specify values for the fields in the query strings of a URL, e.g., @@*jsp?callback=aslHandleAds*, where * represents a wild-card. If we would normalize this string to *jsp?callback=X*, it would not match anymore the previous *exception filter* and thereby we would mis-classify the corresponding URL. Hence, we take care not to overwrite the values in the query strings that appear in the filter lists.

Finally, recall that *Adblock Plus* also includes functionality to block text advertisements that are included in the HTML itself. Quite a number of HTML documents embed these advertisements, which means that the browser extension will not block the associated request because blocking it would also imply blocking non-ad content. Instead, the browser plugin hides this ad content during the rendering phase of the page, in a process called *element hiding*. Since we cannot and do not have access the packet payload we cannot parse the Web page's content and thereby, can neither detect nor comment on this type of advertisements.

## 3.2 Identification of ad-blocker users

To identify if an end user has installed an ad-blocker we rely on the following observation: a browser with such an extension should issue less ad requests than a browser without any ad-blocker. In other words, a low number of ad requests is a strong indicator for the presence of an ad-blocker.

Hence, we have to identify all requests of a particular end user and compute the ratio of ad requests. The ratio for an end user depends on *i)* the browser configuration, i.e., whether there is an ad-blocker installed and, if so which ad-blocker with which configuration, and *ii)* which sites the user visits. Given that many of the most popular sites do indeed have extraneous content [42, 27, 25], the likelihood that an active user visits such a site is substantial. Thus, we can use an active measurement study (see §4) to find ad-ratio thresholds that distinguish between ad-blocker and non ad-blocker users.

Although our primary goal is to identify *any* ad-blocker user, we can additionally rely on plugin updates to identify specific ad-blockers like *Adblock Plus* or *Ghostery*. For instance, the former plugin regularly checks for updates of the filter lists to which the user has subscribed. The update frequency is driven by the expiration time specific to each list, e.g., *EasyList* has a soft expiration date of 4 days [1] and *EasyPrivacy* soft-expires already after a single day [2]. In fact, the *Adblock Plus* contact frequency is quite high: typically upon browser bootstrap or once per day [46]. Hence, monitoring connections to *Adblock Plus* servers is a good indicator for the presence of *Adblock Plus*. To identify *Adblock Plus* servers in the traces we rely on multiple DNS resolvers to obtain an up-to-date list of *Adblock Plus* server IPs.

# 4. METHOD EVALUATION & VALIDATION

We complement our passive analysis with an active measurement study that has two goals *i)* validate that our methodology can classify ad traffic, and *ii)* identify ad-ratio thresholds to differentiate between end users that browse the Web with an ad-blocker and those who do not.

Accordingly, we instruct a popular Web browser to fetch Web sites using different configuration modes, e.g., with *Adblock Plus* enabled or disabled. In parallel we capture the browsers network traffic with *tcpdump* and apply the above methodology to the passive traces.

## 4.1 Active measurement setup

We instrument the widely used browser Chromium with Selenium [20] to crawl the Alexa top 1000 sites. We run the browser in a virtual frame-buffer on a dedicated GNU/Linux machine connected to a university campus network. For each URL in the Alexa top list we start a new browser instance with an empty cache, wait 5 seconds before starting a *tcpdump* traffic trace, load the URL and wait another 5 seconds before closing the browser and *tcpdump*. For each URL we repeat this process 7 times, once for each of the following browser profiles:

**Vanilla:** We do not activate any plugin.

**AdBP-{Ads|Privacy|Paranoia}:** We activate the *Adblock Plus* plugin and configure three separate profiles using the following lists i) the *EasyList* and *non-intrusive advertisements* (Ads), ii) *EasyPrivacy* (Privacy), and iii) *EasyList* and *EasyPrivacy* (Paranoia).

**Ghostery-{Ads|Privacy|Paranoia}:** We activate the *Ghostery* plugin and configure three separate profiles, which block the following object categories i) Advertisements (Ads), ii) Privacy (Privacy), and iii) all categories, including Analytics, Beacons and social media widgets (Paranoia).

| Browser Mode | #HTTPS | #HTTP | #EL$_{hits}$ | #EP$_{hits}$ |
|---|---|---|---|---|
| **Vanilla** | 7,263 | 57,862 | 4,738 | 4,807 |
| **AdBP-Pa** | 4,287 | 48,599 | **6** * | **6** * |
| **AdBP-Ad** | 5,254 | 53,435 | **10** * | 4,279 |
| **AdBP-Pr** | 5,189 | 55,717 | 3,627 | **7** * |
| **Ghostery-Pa** | 2,908 | 48,765 | 940 | 624 |
| **Ghostery-Ad** | 5,734 | 57,425 | 1,326 | 4,668 |
| **Ghostery-Pr** | 6,902 | 55,394 | 4,514 | 2,865 |

**Table 1: Active measurements: Aggregate results for the Alexa top 1K list. Browser modes include Paranoia (Pa), Ad-blocker (Ad) and Privacy (Pr). Classification of URLs based EasyList (EL)and EasyPrivacy (EP). Ad-blockers lessen the total number of requests and lower the ratio of ad requests.**

This experiment results in seven sets of passive traces for each of the top-1000 Alexa Web sites corresponding to the seven different browser profiles viz., with and without *Adblock Plus* and/or *Ghostery* enabled. With this information—the configuration of the Web browser that produced the network trace—we can apply our methodology to each set of traces and, thereby, assess the accuracy of our methodology.

## 4.2 Impact of ad-blockers on traffic

As described in §3, we use *Bro* HTTP analyzer to extract the HTTP information from the set of traces collected during the experiment. We then classify the requests using *libadblockplus*. Table 1 summarizes the total number of HTTP(S) requests in the traces and the corresponding classification of requests according to the filters of *EasyList* and *EasyPrivacy*.

Our first observation is that ad-blockers indeed significantly reduce the number of HTTP and HTTPS requests. For instance, in the *AdBP-Paranoia* mode the browser issues 9K less HTTP requests than in the *Vanilla* mode. The number of HTTP requests issued by a browser configured with *Adblock Plus* in the most aggressive mode is roughly 80% of the corresponding value for the *Vanilla* mode. In this context, 6% of the ad requests in the trace for the *Vanilla* mode either match a filter in *EasyList* (8.1%) or one in *EasyPrivacy* (8.3%). These observations are consistent with those reported in related work, e.g., see [42, 27]. The number of HTTPS connections follows likewise this trend, i.e., 2.9K connections less. This implies that browsers and servers also exchange ad traffic over HTTPS, a case not covered by our methodology.

The number of requests for both plugins configured in the *Paranoia* mode also differs. We remark that the numbers reported in Table 1 are subject to configurations and do not strictly reflect the actual filtering performance of each plugin; rather, they reflect the existence of the filtering process.

Nonetheless, Table 1 shows that the number of objects classified as ad requests is a strong indicator for the presence of an ad-blocker. As expected, ad-blockers hinder many requests from being issued and thus, the number of identified ads with our methodology is small (indicated in bold numbers in the table). In the absence of an ad-blocker, the number of ad requests is significantly larger (see the vanilla browser configuration row).

We note that our approach mis-classifies a small number of requests. We indicate *false positives* with a $*$ in Table 1. *False positives* are requests that the *Adblock Plus* browser plugin does not block but our methodology classifies as ad-related objects. We manually investigate these cases and offer the following explana-

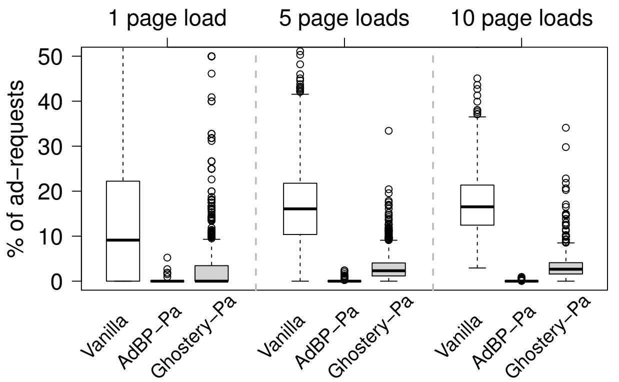


**Figure 2: Active measurements: ratio of ad-requests per browser configuration. Comparison among 1K iterations of 1, 5, 10 randomly selected page loads. The presence of an ad-blocker is more evident when the user becomes more active.**

tion: some are due to inconsistent `Content-Type` values in the responses. For instance, in one case *Bro* reports the MIME *text/x-c*. Our methodology maps this request to the content type `object`, but a manual inspection of the object reveals that this object is in reality a JavaScript object. Remapping the content type to `script` triggers an exception filter that prevents the mis-classification. In fact, the main source of mis-classifications are URLs to JavaScript objects where the `Content-Type` field is set to *text/html*. Modern browsers circumvent this problem since they infer the type of the object without relying on the HTTP headers. Our methodology has to rely on the HTTP headers and thereby is affected by such inconsistencies.

### 4.3 Identifying ad-blocker users

Table 1 illustrates the effectiveness of our approach for classifying ad requests in a network trace in a fashion similar to an ad-blocker. Next, we tackle the question from §3.2: what ratio of ad to non-ad requests is a useful indicator to infer ad-blocker usage? To answer this question we show in Figure 2 a series of box-plots for the ratio of ad requests across browser configurations (for space reasons we only include the results for *Vanilla*, *AdBP-Pa* and the *Ghostery-Pa* configuration modes). For each of these three modes we execute three experiments. We randomly select 1, 5 and 10 sites of the Alexa top 1K list, and compute three ratios of ad requests. Our motivation for choosing different number of sites is to represent users with different levels of activity. We repeat this 1K times. When comparing the box plots we can see that the ad-ratios differ significantly if the number of page loads is sufficiently large, i.e., when users are active. Accordingly, we use a discrimination threshold of 5% when the number of requests is sufficiently large, e.g., 10 page loads or 1K requests. Using a slightly higher or lower threshold does not alter the results significantly.

## 5. DATA SETS

We had access to two anonymized traces from a Residential Broadband Network (RBN) of a large European ISP. These two traces were collected within two customer aggregation networks in the same city. The first trace was captured at a low level which carries the traffic of about 7.5K DSL customers to the Internet. The second trace was captured at the next higher level, which carries the traffic of about 19.7K DSL customers. The up-link speeds are 3 and 10 Gbps, respectively. Table 2 reports the dates when the traces were collected.

| **Trace** | **RBN-1** | **RBN-2** |
|---|---:|---:|
| **Date** | 11$^{th}$ Apr. 2015 | 11$^{th}$ Aug. 2015 |
| **Time** | 00:00 | 15:30 |
| **Duration** | 4 days | 15 and ½ hours |
| **Subscribers** | 7.5K | 19.7K |
| **HTTP**$_{bytes}$ | 18.8T | 11.4T |
| **HTTP**$_{reqs}$ | 131.95M | 85.09M |

**Table 2: Passive measurements: Data sets**

The monitoring infrastructure uses Endace DAG network monitoring cards [11]. These cards support a port-based classification, which is appropriate for HTTP(S) traffic (see [44, 51]). Hence, HTTP traffic can be associated to TCP traffic from (or to) port 80. Likewise, HTTPS traffic relates to *EasyList* downloads can be associated to TCP traffic from (or to) port 443 from (or to) IPs in the list of *Adblock Plus* servers that host these lists. We obtained this list with active measurements (see §3.2) before and after the trace was captured. They did not exhibit differences.

Like many other ISPs, most of the customers of this ISP have a home gateway at their premises that performs Network Address Translation (NAT). These gateways multiplex many user devices, and thereby also browsers, to a single IP address. To report the prevalence of ad-blockers in residential broadband networks we have to identify unique devices and more specifically Web browsers. Maier et al. [45] showed that a good indicator for separating HTTP traffic from multiple devices behind the same NAT is the HTTP `User-Agent` strings of the browsers (in contrast to the strings used, e.g., by software update tools or media players). The rationale behind it is that the `User-Agent` string includes information about the operating system, browser version, etc. Thus, we use the pair end-host `IP` and `User-Agent` to separate our data by end device. Given that most ISPs assign addresses to their customers dynamically, we can only associate an IP address to a household for traces with short duration.

Table 2 gives an overview of the two traces collected in this ISP. The first data set corresponds to a 4-day long trace of the smallest set of customers for which we only capture HTTP traffic, i.e., 7.5K . The second trace is a shorter trace, but it captures peak time traffic for a larger number of end users, i.e., 19.7K . In the remainder of this work we use the latter trace to elaborate on ad-blocker usage, and the former to describe general characteristics of ad traffic.

We pay careful attention to respect and preserve the privacy of end users in our study. First and foremost, we process, aggregate, and analyze the data on a private and secured infrastructure. Second, the IP addresses of the end users are anonymized at the time of the packet capture, i.e., the real IP addresses of the end users were never stored to disk and are unknown to us. Third, we automate the ad classification process which, when completed, truncates every URL in the logs to a fully qualified domain name (FQDN), thereby removing sensitive information.

## 6. AD-BLOCKER USAGE IN THE WILD

Using the methodology from §3 we can proceed to assess ad-blocker usage in the residential broadband network trace `RBN-2`. To infer if an end user[1] is using an ad-blocker, we use the following two indicators: i.e., *i)* the ratio of ad to non-ad requests, and *ii)* automatic *EasyList* downloads by *Adblock Plus* (see §3.2).

[1] We use term "user" to refer to the pair formed by an `IP` address and a browser's `User-Agent` string.

To this end, we have first to identify browsers in the trace. Along with browsers, residential broadband networks indeed manifest a rich and complex mix of other HTTP-based applications. To illustrate this mix, we show in Figure 3 the total number of retrieved HTTP objects vs. the number of retrieved ads[2,3] for each IP address and User-Agent pair on a log-log scale. Due to the large number of data points, we use a heat map to capture the density within each area of the plot. We find in total 508.7K tuples of IP and User-Agent in RBN-2. We see the whole range: some pairs only account for a few requests while others issue orders of magnitude more requests. Overall, we observe 18.89% ad requests in this data set.

Most relevant for us is that there is a substantial number of pairs that request many Web objects but hardly any ads. These are the ones in the lower right hand side of Figure 3. They are most likely browsers that have an ad-blocker installed or visit only sites without advertisements.

However, there are many more points in this plot than we would have expected given the number of monitored DSL-lines (19.7K different households); more than 25 different User-Agent strings per household in average. Upon closer manual inspection we find pairs that identify devices like consoles and Smart TVs. On the other hand, given that most modern end-user devices run many HTTP-based applications in parallel, we also find pairs that correspond to desktop-based gaming applications or mobile apps (these applications use custom User-Agent strings). Since advertisements typically appear within Web sites and mobile applications, we can discard User-Agent strings that do not correspond to this type of applications. Moreover, since in-app ads differ substantially from browser ads, we limit our analysis to Web browsers in this paper. Namely, we restrict our analysis to sessions for which we can associate the User-Agent either to a well-known desktop browser or to a mobile device browser. Indeed, a manual inspection of the pairs shows that some of the points on the right hand side of Figure 3 correspond to mobile apps, which we do not want to consider. Thus, we next use the User-Agent strings to identify popular Web browsers.

## 6.1 Annotation of active users

Our approach is to manually label the User-Agent strings in a subset of the data. We use this as a starting point to subsequently classify the entire data set. Our starting subset is the active users. More precisely, we select tuples that issue more than 1K requests (the heavy hitters) e.g., those corresponding to a few page retrievals. As a result, we obtain a more tractable set of 15.2K pairs, with 1.6K unique User-Agent strings. Among this set of strings, we are able to manually annotate 601, which appear in 9.6K of the heavy-hitter tuples.

With this set of annotated strings, we proceed to classify browsers in the entire set of pairs. We identify 44.1K additional browsers. All these browsers generate 57.2% of the observed requests and 82.2% of the ad requests in the RBN-2 trace. As expected, the heavy hitters issue most of these requests, i.e., 50.6% requests and 72.5% ad requests. We thus continue our study with the set of heavy hitters, i.e., the most active browsers.

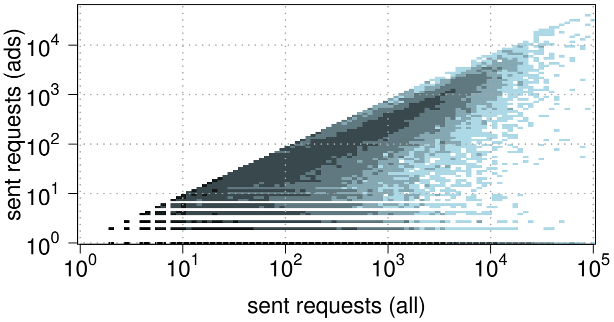


**Figure 3: RBN-2 heat map: Number of total requests (x-axis, log-scale) vs. ad-requests (y-axis, log-scale) per IP, User-Agent pair. Darker points indicate more pairs. Most pairs issue a significant number of ad requests. Other pairs issue many requests but just a few ad requests.**

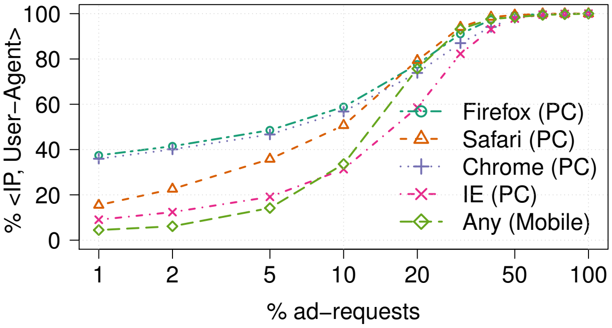


**Figure 4: RBN-2 ECDF: Percentage of ad-requests (x-axis, log-scale) per active browser (those sending more than 1K requests). Many of the Firefox and Safari instances have a low ratio of ad requests.**

We separate these browsers into *i)* mobile and *ii)* desktop versions. We identify 1.9K browsers in the mobile category. They correspond to iPhone and Android phones. These browsers issue 5.9% of both the ad and the total number of requests in the RBN-2 trace. The desktop category accounts for the rest of requests and it is constituted by 7.7K browsers which we further separate into Firefox (3,423), Chrome (2,267), Internet Explorer (654) and Safari (1,324) browsers.

## 6.2 Inferring ad-blocker usage

With this set of annotated browsers, we continue our analysis of ad-blocker usage at our vantage point. We use two indicators: (a) ratio of ad requests, which applies to any ad-blocker, and (b) HTTPS connections to check for filter updates, which is specific to *Adblock Plus*.

**First indicator: low ratio of ad requests.** We use this indicator to detect the presence of an ad-blocker from the HTTP requests. Our motivation for this indicator stems from the insights of the active measurement study (see §4). The rationale is that users that installed an ad-blocker retrieve significantly smaller number of ads than those that have not.

Figure 4 shows the empirical cumulative distribution function (ECDF) of the percentage of ad requests per browser for the annotated set of active browsers. Since not all filter lists are installed by default, e.g., the *EasyPrivacy* list, we only consider ads classified by *EasyList*, which indeed is installed by default. In this context, 40% of the Firefox and Chrome active browsers issue less than 1% ad-requests; they qualify as ad-blocker candidates. By

[2] We use term “ad”, “ad object” and “ad request” interchangeably in this paper to refer to a request that is blacklisted by *EasyList* and its derivatives, as well as those that are blacklisted by *EasyPrivacy*. We also use this term to refer to requests that are whitelisted by the list of *non-intrusive advertisements*.

[3] Note that fetching and displaying an advert can involve several executions and thereby multiple requests [25].

contrast, only for 18% of the Safari and 8% of the IE instances the ratio of ad requests is below the threshold. This can be due to the fact that installing and ad-blocker like *Adblock Plus* in these browsers is a bit more cumbersome and thus might deter their usage. Based on these observations and the results from the active measurement study (see §4), we set the threshold for identifying ad-blocking browsers to 5%, to *i)* tolerate mis-classifications due to content type (see §4.2), and *ii)* take into consideration users that disable blocking for specific sites.

**Second indicator: downloads of filter lists.** As mentioned above, *Adblock Plus* frequently checks for updates to the filter lists. These updates typically occur during browser bootstrap or when a list soft-expires (see §3.2). Thus, we can estimate the number of *Adblock Plus* users by monitoring *EasyList* downloads. However, the *Adblock Plus* browser extension uses HTTPS to download the lists. Hence, we cannot differentiate between multiple browsers hidden behind a single IP (e.g., in the presence of a NAT), since the `User-Agent` is not visible. Instead, we can only report the number of households in which there is at least one device with an *Adblock Plus* installation.

We find several thousands of HTTPS connections to *Adblock Plus* servers in the `RBN-2` trace. These connections are issued from 19.7% of the households in this data set. This number is slightly larger than what has been previously reported. Metwalley et al. [46] reports that the fraction of households with at least one device using an *Adblock Plus* plugin is between 10% and 18%. As stated previously, this information is not sufficient to discern browsers that likely run an ad-blocker, but we can leverage it as indicator and correlate it with the ad-ratio indicator.

**Correlation of indicators to assess *Adblock Plus* usage.** We proceed to correlate the ad-ratio with the *EasyList* downloads indicator. We obtain four classes as the cross product corresponding to a combination of the two indicator values. We summarize these 4 classes along with their corresponding traffic statistics in Table 3. We find that 46.8% of the 9.6K active browsers neither classify as an ad-blocker candidate nor contact *Adblock Plus* servers (denoted in the table with the type name $A$). The orthogonal case, i.e., those annotated with type $C$, constitute 22.2% of the active browsers population. As expected, this class is dominated by Firefox and Chrome browser `User-Agents`. These two types of browsers represent 51% and 32% of the occurrences respectively. Safari on the other hand accounts for 11%. In practice 31% of the Firefox and Chrome instances fall into this class and thereby they probably have *Adblock Plus* installed. This share is slightly larger than the official statistics [18, 15]. This difference could be explained by a permissive threshold value or by a bias due to the population at our vantage point.

There are many browsers for which the download information and the ratio of ad requests produce inconsistent outcomes (types $B$ and $D$). Type-$D$ browsers exhibit ad-blocker-like behavior, albeit they did not attempt to download *EasyList*. Browsers in this category sum up to 15.3% of the active browser population. There are two possible explanations for this apparent inconsistency. The end users might *i)* have installed a different plugin, e.g., *Ghostery* or *NoScript*, or *ii)* have requested content from sites with few advertisements. Based on the observation that other ad-blockers are much less prevalent than *Adblock Plus* [46], we speculate that the most likely cause for this inconsistency is the latter scenario.

The second type of inconsistency relates to type-$B$ browsers. They add up to 15.7% of the active browser population. For these instances we observe an *EasyList* download but the ratio of ad requests is higher than 5%. The most plausible explanation for this contradictory information is that there may be many users in the same household, some of them using *Adblock Plus* and others not.

| Type | Ratio | EasyList | Instances | % requests | % ad reqs. |
|---|---|---|---|---|---|
| **A** | X | X | 46.8% | 22.5% | 46.3% |
| **B** | X | ✓ | 15.7% | 8.1% | 15.8% |
| **C** | ✓ | ✓ | 22.2% | 12.9% | 6.5% |
| **D** | ✓ | X | 15.3% | 7.1% | 4.0% |
| **Any** | - | - | 9.6K | 50.6% | 72.5% |

**Table 3: Ad-blocker usage: classification and statistics for the annotated set of active browsers using the indicators *i) Ratio:* low ratio of ad requests, i.e., $\leq 5\%$, and *ii) EasyList*: HTTPS connections to an *Adblock Plus* server.**

## 6.3 Adblock Plus configurations

There are various filter lists with different objectives available for *Adblock Plus* users. Most notably, there are lists to *i)* block and hide adverts (e.g., *EasyList*), *ii)* whitelist adverts (list of *non-intrusive ads*), and *iii)* protect user privacy by blocking Web trackers (e.g., *EasyPrivacy*). To further elaborate on ad-blocker usage we dissect our corpus of adverts by the list that triggered the classification.

Our first observation is an unexpected high share of ad requests among the likely *Adblock Plus* users (type-$C$). For this observation, we refer to Table 3, which reports the contribution of each user category to the total number of ad requests in `RBN-2`. One would expect the relative share of ad requests for *Adblock Plus* users to be close to 0%. Instead, we find a share of 6.5% ad requests. The explanation for this high percentage is that our threshold-based classification is based on *EasyList* hits. We use this list because it is by default activated upon *Adblock Plus* installation; along with the list of *non-intrusive* advertisements whitelist. However, the numbers that we report in Table 3 correspond to all hits, including those triggered by the other lists, i.e., *EasyList* derivatives and *EasyPrivacy*. Indeed, we observe that 82.3% and 11.1% of the positive classifications for *Adblock Plus* users relate to filters in *EasyPrivacy* and in the list of *non-intrusive ads* respectively. This observation motivates the consequent analysis about the lists that *Adblock Plus* users subscribe to.

**EasyPrivacy.** Metwalley et al. [46] report that 77% of the users contact a tracker immediately after they start browsing the Web. We can leverage this observation to estimate the extent to which (if at all) an *Adblock Plus* user interacts with a tracker. Our assumption is that such interactions should only occur for those users who do not install the *EasyPrivacy* list. We observe that only 0.1% of the *non-adblock* users do not issue requests matching *EasyPrivacy* filter rules, i.e., almost every user contacts a tracker. By contrast, the corresponding fraction for *Adblock Plus* users is 5.1%. If we use a more permissive value, i.e., 10 requests to account for misclassifications, then the percentage of *Adblock Plus* users that likely installed *EasyPrivacy* is 13.1%. Overall, we see a consistent difference around 15% using different values.

In light of these observations, we speculate that most *Adblock Plus* users, i.e., more than 85%, do not subscribe to *EasyPrivacy* but just to *EasyList*. In fact, this argument is supported by a blog post from the *EasyList* maintainers, which reported back in 2011 that only 4.1% of their 12 million users subscribed to *EasyPrivacy* [9]. The key take-away is that it seems that *Adblock Plus* users install this software to block annoying advertisements but do not configure it to protect their privacy.

**Non-intrusive adverts.** We conduct a similar analysis to elaborate on the prevalence of the *non-intrusive ads* list. We find that *Adblock Plus* users issue 7.9% of the total number of whitelisted requests. In comparison, *non-adblocker* users generate 37.9% of this type of requests, despite the fact that the population of this set is almost twice the population of the former set. This first observation highlights the importance of this list for advertisers and publishers alike, i.e., *Adblock Plus* users still generate a significant fraction of ad requests.

When trying to elucidate whether an *Adblock Plus* user opts out of this whitelist, we find that 11.8% of the *Adblock Plus* users issue no requests of this kind. The corresponding percentage for *non-adblock* users is 6.1%. The reason why many *non-adblock* users do not issue ad requests of this kind is because this type of adverts are less prevalent (see §7). At less than 10 issued whitelisted requests, the difference between both groups is roughly 20%, a pattern that is repeated across different values. This is, perhaps, an indication that at most 20% of the users actually deactivate this list, thus disabling the whitelisting of adverts that conform the non-intrusive ads guidelines. We emphasize a word of caution on this last statement and remark that corroborating this observation would require to conduct an analysis of the browsing patterns of *Adblock Plus* users.

**Summary:** Our main observation is that a significant fraction of the most active users in our traces browse the Web with *Adblock Plus*, i.e., 22.2% of the active users. This extension is especially popular among Chrome and Firefox users, i.e., 30%; and less popular among Safari and Internet Explorer users. We find that most *Adblock Plus* users do not install the *EasyPrivacy* list that protects them from trackers. Hence, it seems that *Adblock Plus* users are mostly interested in blocking ads rather than protecting their privacy. However, this might be an awareness problem. We also find that most *Adblock Plus* users probably do not opt out from the list of acceptable ads (*non-intrusive ads*) which is enabled by default. In fact, we find that the set of heavy-hitter *Adblock Plus* users still generates a substantial number of such ads, even when compared to *non-adblock* users. This observation suggests that conforming to the acceptable ads guidelines may benefit some players in the advertising domain.

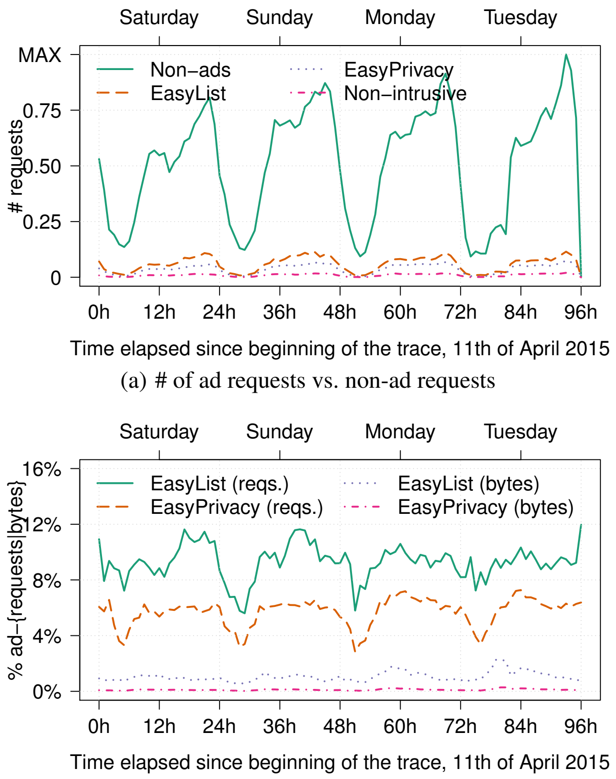
# 7. ADS IN THE WILD

One under-explored aspect of the ad-scape is the prevalence of ads that end users experience during *regular browsing sessions*. Most previous work (see e.g., [25, 42]) has relied on active measurements and therefore cannot capture how the average user interacts with the *ad-scape* while browsing the Web. Thus, in this section we use our passive measurement methodology to study basic properties of ad traffic in the wild. Concretely, we investigate *i)* basic ad-traffic properties, *ii)* content-related properties of the observed ads, and *iii)* whitelisted ads.

## 7.1 Ad-traffic characterization

To comment on the temporal characteristics of the ad-traffic we conduct our analysis on the `RBN-1` trace, as it spans a longer period than `RBN-2`. We find that 17.25% and 1.13% of the requests and bytes respectively correspond to ad objects in the `RBN-1` trace. To put these numbers into perspective, we refer to our active measurement results (see Section 4). For the Alexa top 1K sites we classified 16.4% of the total requests as adverts (see Table 1).

We highlight the variability of ad traffic in Figure 5(a), where we depict a time series for the number of requested Web objects vs. the number of ad requests using bins of 1 hour. The non-ad requests manifest the characteristic time of day and of week pattern of residential networks. During the night there are relatively fewer requests and the busy time is in the evenings, right before midnight. On the weekend there are fewer requests than during the week, in particular on Saturday. Moreover, the lunch break is also clearly visible. Surprisingly, the ad-related requests do not show the same pattern. To visualize these differences we plot in Figure 5(b) the percentages of ad requests and ad bytes over time. Here we only consider ads reported by filters in *EasyList* and *EasyPrivacy* (excluding *non-intrusive ads*). The figure reveals that, surprisingly, the ratio of ad requests also manifests a diurnal pattern, ranging from 6% up to 12%, instead of having a constant rate.



(a) # of ad requests vs. non-ad requests

(b) % of ad requests and ad bytes according to *EasyList* and *EasyPrivacy*

**Figure 5: Time series for ad and non-ad traffic (1h time bins, `RBN-1` trace). Ad traffic exhibits a daily pattern, both in terms of number of requests and in terms of its ratio to all requests in the trace.**

To explain this surprising diurnal pattern we offer two possible explanations. The first one is that users request different content and that the pages that serve this content have a different ratio of advertisements. For example, streaming video chunks might result in a very low ratio of ads. Another example are Web site categories, e.g., the *news* category has more objects than other categories (see [27] for a detailed study about the complexity of popular Web sites). Our second explanation for the diurnal pattern is that the share of ad-blockers users varies at different times of the day. We leverage the classification described in §6.1 to investigate this for `RBN-2`. Our finding is that at peak time the number of *non-adblocker* active users is twice the number of active *Adblock Plus* users. By contrast, during the off hours the number of active *Adblock Plus* and *non-adblocker* users is roughly the same.

| **Content-type** | **Ads** | | **Non-Ads** | |
|---|---|---|---|---|
| | Reqs | Bytes | Reqs | Bytes |
| `image/gif` | 35.1% | 14.1% | 3.5% | 0.7% |
| `text/plain` | 28.7% | 34.2% | 14.3% | 2.1% |
| `text/html` | 14.4% | 11.8% | 7.6% | 1.1% |
| - | 11.8% | 5.4% | 28.7% | 63.4% |
| `app./xml` | 4.5% | 2.2% | 1.8% | 0.2% |
| `image/png` | 1.9% | 1.6% | 5.1% | 0.8% |
| `image/jpeg` | 1.8% | 5.2% | 19.8% | 3.4% |
| `app./x-shock.` | 1.4% | 8.1% | 0.2% | 0.1% |
| `video/mp4` | 0.0% | 10.9% | 0.3% | 8.6% |
| `video/x-flv` | 0.0% | 5.4% | 0.1% | 3.1% |

**Table 4: Trace `RBN-1`: ad traffic by *Content-type*.**

Our next question relates to the type of adverts that we see. To answer it we dissect the ad requests by the list that triggers the classification. *EasyList* causes significantly more hits than *EasyPrivacy*, which in turn triggers more classifications than the *non-intrusive ads* list. More precisely, *EasyList* classifies 55.9% of the ad requests in `RBN-1`. The *EasyPrivacy* is responsible for 35.1% of the requests. The *non-intrusive ads* list triggers the remaining matches. We observe the same trend in `RBN-2`.

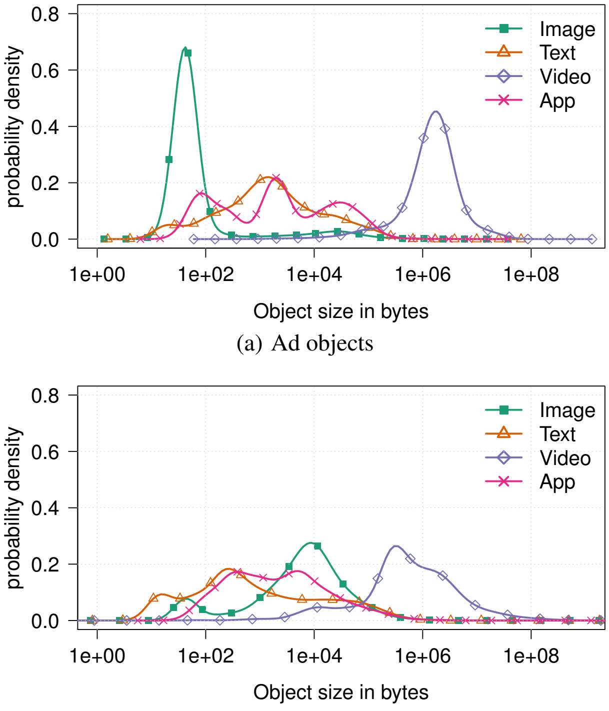
## 7.2 Ad-related objects by Content-Type

Next, we consider the types of the requested ads (e.g., image vs. video ads) and their prevalence. Therefore, we analyze the *Content-Type* of the Web objects in `RBN-1`. Table 4 shows the most prevalent objects according to the MIME type reported by *Bro* HTTP analyzer along with their corresponding contribution to the total traffic and to the ad traffic in terms of requests and bytes. Most ads are either *image/gif*, *text/html*, or *text/plain*. The fraction of bytes is dominated by *text/html* objects, while the fraction of bytes for the type *image/gif* is relatively small. The latter is not surprising given that many of the ad objects used to track users are small, i.e., 43 bytes. At the other extreme are videos (i.e., *video/mp4, video/x-flv*), flash objects (i.e., *x-shockwave*), and non-*gif* images such as *image/jpg*. All these types contribute a higher fraction of bytes than requests.

To highlight the different distributions, we show in Figure 6(a) the density of the ad-object size on a log scale separated by the `Content-Type`. We consider four different classes: images (*gif*, *jpeg*, and *png*), text (*html*, *plain*), video (*mp4*, *flv*), and applications (*xml*, *flash*). The density of the logarithm highlights that most images are very small (43 bytes), while most videos are rather large ($>$ 1 MByte). If we compare these values to the typical object size of non-ad objects, see Figure 6(b), we notice significant differences. Most non-ad videos are smaller than ad videos, while most non-ad images are larger. One of the reasons is that most video-streaming providers split regular videos into multiple chunks and each chunk corresponds to one Web object. Since most video ads only last for 15-45 seconds and advertisers expect end users to see the complete video stream, chunking may be considered to be unnecessary. Moreover, most ad videos typically have a length in the same order of magnitude. Surprisingly, we see that non-ad text objects are likely to be smaller. These are likely requests involving high-interactive sites, e.g., for auto-completion or for suggestions.

## 7.3 Non-intrusive advertisements

Next, we look at the relevance of the list of *non-intrusive ads*. On the one hand, ad-blockers threaten the financial backbone of the Web. On the other hand, they also ensure some balance by preventing ads from becoming too intrusive [19]. The result is the *non-intrusive ads* list which whitelists adverts and is enabled by default. Indeed, according to Financial Times [6] some companies including Google, Microsoft and Amazon, pay money to *Adblock Plus* to be whitelisted. But what is the effect of this list? Given our vantage point and our ability to classify traffic the same way *Adblock Plus* does, we can investigate the impact of the whitelist using the `RBN-2` trace.



(a) Ad objects

(b) Non-Ad objects

**Figure 6: PDF of the object size distribution of the requests according to their MIME type (`RBN-1`). Ad-related objects exhibit characteristic sizes.**

We first ask how many of the ad-related requests match the whitelist. This is the case for 9.2% of the ad requests. While this number may seem low at first, we calculated it using also the hits triggered by an *EasyPrivacy* filter. If we restrict ourselves to ad requests identified by *EasyList* and *non-intrusive ads*, then the percentage that is subject to whitelisting grows up to 15.3% viz., these adverts would not be blocked by the default *Adblock Plus* installation.

**List accuracy.** However, these numbers likely *overestimate* the real impact that this list has. We manually inspected the filter rules and found some anomalies: some rules are overly general, e.g., they whitelist an entire domain rather than specifically addressing ad-related parts. For example, many requests match the `@@||gstatic.com^$document` filter rule, which whitelists the entire `gstatic.com` domain. This domain hosts unsuspected fully qualified domain names (FQDNs) e.g., `fonts.gstatic.com` and services such as *Street View*. Hence, this filter list may also whitelist non-ad-related traffic. On the other hand, and referring to the previous example, font objects may very well be necessary to display an advertisement e.g., *Google's AdSense*.

Therefore, we ask how much of the whitelisted traffic would have been otherwise blocked by a blacklist, i.e., when *Adblock Plus* users choose to stop allowing *non-intrusive* advertisements. The ratio is surprisingly small: only 57.3% of the whitelisted requests would have been blacklisted. Moreover, 23.2% of those would be filtered by *EasyPrivacy*. These observations highlight that the list of *non-intrusive ads* should be handled with caution when trying to identify adverts. In the remainder of this subsection we consider only those whitelisted requests which "match the blacklist".

**Publishers.** Recall, we can identify the main page that originated the request using the methodology described in [56]. We find 991 unique FQDNs with more than 1K blacklisted requests to which we can associate 84.0% of the total requests blacklisted by *EasyList* and its language derivatives. The *non-intrusive ads* list whitelists 8.6% of these requests.

We find that some sites in the *dating*, *shopping*, *translation*, *audio and video streaming* categories, as well as some sites in the *mixed content* categories benefit the most from the whitelist[4]. On the other side, we find that most of the sites without whitelisted requests belong to the *adult content* category, followed by Webs in the *mixed content* category. Here, we also find another one in the *file sharing / video streaming* category. It is not surprising to see that these sites are not whitelisted. However, it is surprising to find a few instances of popular sites in the *news* category (in fact they appear in the Alexa top 1K). Thus, an *Adblock Plus* user would block every ad-related request related to these sites. We manually checked these sites with *Adblock Plus* to corroborate this observation.

**Ad-tech companies.** We repeat the same analysis to shed light on how the *non-intrusive ads* list affects ad-tech companies. We select FQDNs with more than 10K blacklisted requests. These domains sum up to 82.1% of the total blacklisted requests, from which 11.1% are whitelisted.

Like in the publisher case, we see a mix. The list of *non-intrusive ads* whitelists 47.9% of Google's requests (recall we exclude HTTPS traffic). Some of its services do get most of their requests through (e.g., analytics, ad-services) while others do not. While the bulk of these requests corresponds to Google, we also see other companies benefiting from the *non-intrusive ads* list. One particular example is a *technology/Internet* Web site that operates its own ad-platform, for which the *non-intrusive ads* list whitelists 94% of the otherwise blacklisted requests.

**Summary.** Our main observation is that a significant share of the requests and bytes in a residential broadband network are due to online advertising e.g., 17.25% and 1.13% of the requests and bytes respectively for trace `RBN-1`. We observe that ad-related traffic exhibits a different diurnal pattern than regular traffic. Moreover, ad objects have more characteristic sizes than non-ad objects. In ads *gif* images dominate in terms of number of requests, followed by *text/plain* objects, which in turn dominate the share of ad traffic in terms of bytes.

Moreover, our analysis of the potential benefits of whitelisting i.e., the list of *non-intrusive ads*, shows that some content publishers substantially benefit from this list, while others not. Among the latter are sites in the *adult* category. However, we also find popular sites in the *news* category among them. Some ad-tech companies also benefit from this list; for instance Google, which carries the bulk of the whitelisted traffic and for which 47.9% of the ad-related traffic is whitelisted. Another example is a popular *technology/Internet* Web site. This site operates its own ad-platform, for which the *non-intrusive ads* list whitelists 94% of the otherwise blacklisted ad traffic.

## 8. ADVERTISEMENT INFRASTRUCTURE

Next, we turn our attention to the infrastructure that serves ad-related objects. This study is motivated by the need to better understand the ad-scape [7], a complex and diverse ecosystem composed by hundreds of companies that provide numerous services and closely interact with each other, e.g., ad-networks and exchanges. In this section we study the server-side infrastructure from the perspective of an end user using *Adblock Plus*. Namely, which infrastructures end users contact, how often and how many, and if we can find signs of real-time bidding.

### 8.1 Server side ad infrastructure

The first aspect of the *ad-scape* infrastructure are the characteristics of the Web servers that deliver ad objects to the end users. We use the term server to refer to an IP address. Note, that any of these addresses may on the one hand be only a front-end of a large server farm or on the other hand a server that is co-located with other virtual servers. We find 29.0K and 19.6K servers in `RBN-1` that serve ad objects according to *EasyList* and respectively *EasyPrivacy*. Some servers (i.e., 5.2K) serve objects matching both lists. As one may expect—like almost every other distribution in the Internet—the distribution of requests per server is heavy-tailed (not shown). If we use only *EasyList* driven classification, the median number of ad objects per server is 7, the mean is 438, and the $90^{th}$ / $95^{th}$ / $99^{th}$ percentiles are respectively 320 / 1.1K / 6.8K ad objects. The busiest server in the `RBN-1` trace, which is operated by Liverail, received 312.3K ad requests in total.

The second aspect relates to the objects served by these servers: do they exclusively serve ad-related objects or do they serve regular content as well? One argument for the first case is that by now there is a separate infrastructure and market dedicated to the ad-tech ecosystem. The opposing argument is that one can take advantage of synergy effects by delivering ads via the same infrastructure as regular content. We find 222.2K servers in `RBN-1`. For 21.1% of them we classify at least one request as an ad object. These IPs serve 54.3% of the total number of non-ad objects in `RBN-1`. However, about 6.9K servers deliver exclusively ad objects. Here, we consider that a server is exclusively dedicated to deliver ad objects if our methodology identifies more than 90% of its requests as adverts. We consider this reasonable since our methodology may not be able to identify all ads and thus provides only a lower bound. We use the previous threshold to find 10.1K ad servers, which altogether deliver 32.7% of the adverts. Likewise, we define the notion of "tracking server" to refer to the set of servers that only serve ad-related objects identified using *EasyPrivacy*. We find 3.3K of these servers, which deliver 18.8% of all the ad-related objects reported by *EasyPrivacy*.

Next, we consider the Autonomous Systems (ASes) which host the ad servers since we want to understand if ad traffic is highly concentrated in a few large infrastructures. To this end we use the global routing information in order to determine the AS that is responsible for the IPs of the servers. The top-10 ASes contribute to the majority of the ad objects in the `RBN-1` trace, namely 56.8%. Among these ASes, see Table 5, we find four categories of players: search engines, cloud providers, CDNs and two ad-tech ASes, i.e., AppNexus and Criteo. Google leads this ranking with 21.0% and 33.9% of all ad requests and bytes, respectively. The relative ratio of ad objects for Google traffic is 50.7% and 15.9% of the total ad requests and bytes to this AS. This ratio may at first seem large and one may ask if our methodology is sound. However, recall that Google switched many of its services to HTTPS—apparently mostly for their "core" content, i.e., search results and video streaming.

[4] We use `http://sitereview.bluecoat.com` to classify Web sites into categories.

| AS | %ads relative to ad objects in trace | | % ads relative to all objects per AS | |
|---|---|---|---|---|
| | Requests | Bytes | Requests | Bytes |
| Google | 21.0% | 33.9% | 50.7% | 15.9% |
| Am.-EC2 | 7.0% | 4.6% | 19.8% | 2.8% |
| Akamai | 6.5% | 19.0% | 6.4% | 1.0% |
| Am.-AWS | 5.5% | 1.1% | 45.9% | 16.6% |
| Hetzner | 3.4% | 1.4% | 23.4% | 3.5% |
| AppNexus | 3.1% | 0.4% | 32.9% | 50.2% |
| MyLoc | 2.9% | 3.0% | 64.0% | 14.9% |
| SoftLayer | 2.8% | 0.4% | 48.5% | 1.9% |
| AOL | 2.7% | 0.3% | 74.7% | 25.4% |
| Criteo | 1.9% | 1.1% | 78.1% | 88.2% |

**Table 5: Trace RBN-1: Ad traffic by AS for top-10 ASes.**

Among the other top contributors are also cloud providers, including Amazon, Hetzner, and MyLoc. Finding cloud providers in the list of top contributors highlights that clouds are also used by the advertisement industry. While some companies opt to manage their own AS to better control and operate their ad infrastructure, others opt to take advantage of the massive amount of resources and flexibility that cloud providers offer. Finally, CDNs like Akamai (which deploys caches within ISPs or within their own address space) and SoftLayer form the third category of ASes that serve ad objects. The presence of CDNs in this list of top contributors supports the argument that the same infrastructure that serves regular content delivers also ad objects.

The last category of ASes in Table 5 includes the relatively unknown ASes AppNexus and Criteo. These are companies whose main business is online advertising. They operate just a few servers (those visible in our traces), e.g., 39 for Criteo. Excluding the landing pages of these companies, we expect these ASes to mostly serve adverts, which is not strictly the case. While we classify 50.2% of the AppNexus bytes as advertisement traffic, the ratio of the requests is much lower (32.9%). Hence, our methodology probably underestimates the number of ad requests for this AS. One possible explanation for this mismatch is that the lists are conservative and do not address all possible URLs to prevent blacklisting desired traffic. Another possible explanation is that fetching an advert can require several JavaScript executions (see [25]) and thereby involve multiple non-ad requests as well. If we do not classify all requests in this chain as ad objects, except for one, the ratio of adverts per AS may decrease. In contrast to AppNexus, the percentage of ad traffic for Criteo in terms of requests and bytes is as high as 78.1% and 88.2%, respectively.

## 8.2 Ad-exchanges and real-time bidding

Real-time bidding (RTB) refers to the process of *selling* advertisement space on a per-impression basis, i.e., per user. This is a standard process in today's Internet (see [23, 53, 25]). The process works as follows. When a user requests an advert from an ad exchange, this exchange contacts multiple advertisers and the highest bidder among them wins the right to display the ad to the user. Advertisers can use many pieces of information to decide on their bid, including but not limited to geographical location, age, or gender. Usually, exchanges wait for around 100 ms before closing the auction [14]. Earlier work revealed the complexity of the ad-ecosystem which involves multiple players [23, 58, 49].

With our traces we can identify real-time bidding in the wild by leveraging a threshold of 100 ms for answering a request. The key observation here is that the bidding threshold adds extra delay to the *HTTP hand-shake* i.e., the time difference between the first HTTP response and the first HTTP request packet. If the delay is larger than 100 ms it may be due to RTB. However, it may also be due to large network delays. To avoid mis-classifications we use the TCP hand-shake time—the difference between the TCP SYN-ACK and the TCP SYN packets—as a proxy for the network round trip time (RTT) to the server. Note, that given the location of our monitor within the aggregation network, the TCP hand-shake time only captures the wide area delays and thus automatically removes access network variations. Likewise, we remove biases due to the servers' locations viz., otherwise data in Europe can be expected to be served faster than that in the US or even Asia. Finally, if the HTTP object is fetched via a persistent connection we still use the TCP hand-shake time from this connection as the delays usually do not vary that significantly within a few seconds—the expected durations of these persistent connections.

Figure 7 shows the *density of the logarithm* of the difference between the HTTP and TCP hand-shake times for the RBN-2 trace, dissected into ad and non-ad HTTP transactions. The first observation is that most of the hand-shake time differences are small, i.e., 1 ms. They relate to noise on the network path or to the processing overhead at the server to determine the HTTP response. The second observation is that while most of the non-ad objects have a short hand-shake time, namely less than 10 ms, ads have—more often than non-ads—larger hand-shake times. This can be seen from the three clear modes at 1 ms, 10 ms and 120 ms in Figure 7. Compared to the non-ad objects a much larger share of ad objects exhibit a time difference between hand-shakes of more than 100 ms. This suggests the presence of *back-offices*, which include ad-exchanges to enable RTB and CDN to fetch objects from other distant servers [49].

In fact, a manual inspection of the fully qualified domain names with such large hand-shake time differences (≥90 ms) reveals that these host names belong to ad-tech companies. To give an example, Google's DoubleClick, which offers RTB, contributes to 14.5% of the ads in this range. We also find other organizations offering RTB, e.g., Mopub (an RTB exchange for mobile in-app ads), the Rubicon project, Pubmatic or Criteo. Each of these organizations contributes roughly 5% to the total number of ads in this range. We also find Web tracking companies here, e.g., AddThis.

**Summary:** The ad infrastructure is mostly concentrated in a few ASes. Along with the expected players, e.g., Google, we observe that some ad-tech companies opt to either deploy their infrastructure in clouds or rely on CDNs, while others choose to manage their own AS, e.g., AppNexus and Criteo. Moreover, our data suggests that often the same infrastructure serves ad content as well as regular content, although some servers do indeed only serve ad-related content. Finally, we show that ad requests typically involve more back-end functionality, i.e., RTB, than non-ad requests, which again highlights the complexity underlying this ecosystem.

# 9. RELATED WORK

Content and services which are offered for free on the Internet are primarily monetized through online advertisement. A rich body of literature seeks to broadly understand the scale, dynamics, mechanisms, economics and general interest concerns related to advertisements on the Web. In this section we provide a summary of the most relevant lines of research concerning our work.

**Privacy and security aspects.** The first and largest line of research includes studies that investigate the extent to which advertisements violate or are in conflict with end users' privacy [35, 40,

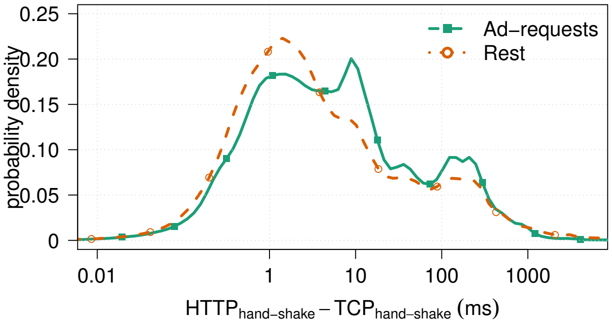


**Figure 7: Real time bidding in RBN-2: difference between the HTTP and TCP hand-shake latencies dissected by type of request. The more pronounced modes at 10 ms and 100 ms suggest that ad requests typically involve more back-end functionality than non-ad requests.**

36]. Other studies propose to address these issues with privacy-aware ad technologies [37]. Security-centric studies report the prevalence and properties of malicious ads [59, 43] and how to detect them [54, 53]. Another related study concerns ad injecting browser extensions [57]. Other studies in this area paid special attention to targeted advertisements. The goal of these advertisements is to improve the effectiveness of the displayed ads to the user by matching the users' interest. Farahat et al. [31] report an empirical evaluation and confirm the effectiveness of such advertisements. However, they also outlined ways in which more sophisticated targeting algorithms can cause harm. A large body of related work investigates privacy issues related to such user profiling, e.g., [35, 40, 36, 37, 30]. Gill et al. conducted a passive measurement study to quantify the information that is aggregators collect and assess, at the same time, the value of this data [32]. Balebako et al. propose tools to mitigate behavioral advertisements [24]. Krishnamurthy and Wills [41] investigate privacy diffusion in a longitudinal study, where they report about the increasing aggregation of user-related information by a few companies, as well as the limitations of existing protection techniques.

**Ad content and traffic characterization.** Another notable line of research concerns the empirical characterization of the online advertisement landscape as well as its impact on Web site complexity. Guba et al. [34] and Barford et al. [25] characterize the ad-scape and highlight its underlying complexity. For instance, how ad exchanges enable real-time bidding to sell advert placements on a per user basis. Pujol et al. [49] report the scale of such "hidden" interactions in a passive measurement study. However, advertising has also a very visible and pronounced influence on today's Web site complexity. Krishnamurthy and Wills [42] report that 25–30% of the Web objects in the Alexa most popular sites are extraneous.

**Mobile ads.** There is one branch of related studies that focuses on advertising in the mobile domain. For example, Rodríguez et al. [55] quantify empirically mobile ad traffic via passive measurements. They reveal insights in the delivery mechanisms and outline means for optimized ad delivery. Targeting mechanisms used in the mobile world (e.g., location-based or user-based targeting) are not rare in today's Internet [26]. In this regard, Nath [47] characterizes the ads displayed in mobile applications and reports about the information that these apps collect for targeted advertising. Moreover, energy consumption is an important topic in the mobile domain. Thus, some studies also devoted their attention to quantify the energy consumed by mobile advertisements [22, 29, 28, 55]. In this regard, energy savings can be achieved by either pre-fetching ads [28], or by blocking them to reduce radio traffic [50].

**Ad-blocking tech and tracking services.** Butkiewicz et al. [27] report the share of extraneous content like ads in Web pages, and conclude that an ad-blocker can reduce the median number of requested objects per site by up to 75%. Guglemann et al. [33] investigate how to detect privacy-intrusive trackers and services from passive measurements. Kontaxis and Chew [39] describe a tracking protection mechanism for the Mozilla Firefox browser. Metwalley et al. [46] contribute to this line of research with a passive measurement study about the extent to which Web trackers follow users in a residential broadband network. One of their findings is that while many users install *Adblock Plus* (roughly 18% of the households), most of them do not install an extension to protect their privacy.

Previous research on online advertisements mainly focused on the empirical classification and characterization of advertisements via active measurement studies. In the passive measurement domain, related work has mostly focused on assessing privacy issues. Our work complements this body of work with a passive measurement study about advertisement traffic and ad-blocker usage in a residential broadband network.

## 10. DISCUSSION

Our methodology leverages *Adblock Plus* functionality to provide insights into how ad-blockers may influence ad traffic dynamics and thereby, Web traffic. Although we show that the proposed methodology is capable of classifying ad requests in HTTP header traces, the classification approach comes at a price.

First of all, the classification of ad requests in header traces is complicated by the lack of structural information in the absence of the HTML payload; individual HTTP requests cannot be associated with Web objects (e.g., if an image is embedded in an iframe). This structural information is leveraged by *Adblock Plus* to improve the ad detection.

To tackle this challenge, we propose a methodology to partially reconstruct the Web page structure. Our methodology mainly employs the construction of a referrer map which associates individual requests (e.g., images, video, CSS, or JavaScript) with the accessed page. While this approach allows us to cluster related requests by page, it cannot reconstruct the entire Web page structure as needed by *Adblock Plus* to achieve higher detection accuracy. We suggest that a complete reconstruction is only possible by accessing the payload and executing the embedded JavaScript code, which might further manipulate the Web page structure.

Moreover, ad and non-ad objects can be transferred over HTTPS, or a mixture of HTTP and HTTPS (e.g., the landing page over HTTPS and the ads over HTTP). Since URLs in HTTPS transfers cannot be analyzed, we cannot always associate all requested objects with a page when constructing the referrer map of such a page. We are, thus, also unable to reason about the prevalence of ad traffic carried over HTTPS connections.

Second of all, our methodology can underestimate the volume of ad traffic in the presence of *hidden ads*, i.e., ads embedded in the main HTML of the page whose retrieval cannot be blocked. To identify these kind of ads we would require access to the entire HTML document (i.e., packet payload), which is not possible in our study. In cases where the payload can be analyzed, our methodology can be extended to detect *hidden ads* and address the challenges discussed above.

Lastly, we remark that there are many other ad-blockers available to end users besides *Adblock Plus*. We may not be able to detect users with different extensions (or even *Adblock Plus* with custom configurations) if they are tailored to block different objects than those affected by *EasyList*.

Furthermore, analyzing the number of ad requests (indicator 1, see §6.2) involves a set of potential biases: *i)* custom configurations and different filter lists (e.g., Web site whitelisting), *ii)* cascaded effects in which the blocked ad request who have triggered subsequent requests, or *iii)* the interaction of *Adblock Plus* with other blocking extensions. The immediate consequence of such biases is that we may *underestimate* the overall usage of ad blocking browser extensions. Moreover, caches and ad blocking middle-boxes or proxies can also decrease the number of observed ad requests. Consequently, confusing *Adblock Plus* instances with ad blocking proxies will lead to *overestimation* of the number of *Adblock Plus* users. To limit the effect of these biases, we introduced the filter list download indicator (§6.2) which monitors updates triggered by *Adblock Plus*. However, since such updates are fetched via HTTPS, we cannot observe the *User-Agent* string required to differentiate different browsers behind a NAT.

We point out that this is a general limitation of the presented *results*, which we produced using the lists mentioned in Section 2, rather than of the applied *methodology*. We selected *Adblock Plus* based on its popularity among end users and note that our approach can be extended to consider other ad-blockers.

# 11. CONCLUSION

The goal of our work is to elucidate the interactions between the end users and the ad ecosystem. We conduct a passive measurement study using traces collected at a residential broadband network of a major European ISP with two objectives in mind. First, elaborate on the prevalence of ad-blockers to inform the ongoing debate regarding these tools. Second, complement related work by characterizing ad traffic at this vantage point using *Adblock Plus* functionality.

Our main observation is that a significant fraction (22%) of the most active users in our traces browse the Web with *Adblock Plus*. Surprisingly, we find little evidence that *Adblock Plus* users install the *EasyPrivacy* list of filters, which aims to protect end users' privacy by blocking trackers. Likewise, our results suggest that most *Adblock Plus* users do not opt out from the list of acceptable ads that is enabled by default in *Adblock Plus*. Based on these observations we conjecture that *Adblock Plus* users are mostly interested in blocking annoying ads rather than protecting their privacy, or that they are not aware of these options or how to change them.

Motivated by these observations, we also investigate the potential benefits of whitelisting, i.e., the list of *non-intrusive ads*. This list can be tremendously beneficial to some content publishers and ad-tech companies given the number of *Adblock Plus* users. We find that 9% of the ad-related requests are whitelisted by this list, while 56% and 35% of the ad-related requests are blacklisted by *EasyList* and *EasyPrivacy*, respectively.

Overall, we find that 18% and 1% (requests and bytes) of the total traffic at our vantage point relates to ad traffic. This share of ad traffic is distributed across different infrastructures, including content and cloud providers and CDNs. However, these infrastructures are concentrated in a few ASes. The top 10 ASes include the expected players, e.g., Google (which dominates the list in terms of requests and bytes). We further observe that some ad-tech companies opt to either deploy their infrastructure in clouds or rely on CDNs, while others choose to manage their own AS, e.g., AppNexus and Criteo. Moreover, our data suggests that often the same infrastructure serves ad content as well as regular content, although some servers do indeed only serve ad-related content.

We also show that ad-related requests typically involve more back-end functionality than non-ad requests. This complexity results in higher observed response times for ad objects than non-ad objects. These latency inflations suggest the presence of real-time bidding within ad exchanges.

This paper takes a first step towards understanding how users employ ad-blockers and what advertisement traffic is potentially affected by this software. Yet, our study also poses new questions. In future work we plan to extend our analysis to better understand the reasons that drive end users to use ad-blockers. Moreover, we also plan to explore the economic impact and implications that *ad-blocking* tech has for the "free" Web.

## Acknowledgments

We would like to thank the anonymous reviewers and Philipp Richter for their constructive feedback. This work was supported by the German Ministry for Education and Research project BBDC (Berlin Big Data Center; funding mark 01IS14013A).

# 12. REFERENCES

[1] https://easylist-downloads.adblockplus.org/easylist.txt.

[2] https://easylist-downloads.adblockplus.org/easyprivacy.txt.

[3] Acceptable Ads Manifesto. https://acceptableads.org/en/.

[4] Adblock Plus Filters Explained. hhttps://adblockplus.org/en/filter-cheatsheet.

[5] Allowing Acceptable Ads in Adblock Plus. https://adblockplus.org/en/acceptable-ads.

[6] Amazon, Google and Microsoft Escape Adblock Plus, for a Price. http://www.engadget.com/2015/02/03/amazon-google-microsoft-adblock-plus/.

[7] Display LUMAscape. http://www.lumapartners.com/lumascapes/display-ad-tech-lumascape/.

[8] DoubleClick Nearing Privacy Settlements. http://news.cnet.com/2100-1023-871654.html.

[9] EasyList Statistics: August 2011. https://easylist.adblockplus.org/blog/2011/09/01/easylist-statistics:-august-2011.

[10] Electronic Frontier Foundation's Privacy Badger. https://www.eff.org/privacybadger.

[11] Emulex Endace DAG technology. http://www.emulex.com/.

[12] EU Carriers Plan to Block Ads, Demand Money from Google. http://arstechnica.com/business/2015/05/eu-carriers-plan-to-block-ads-demand-money-from-google/.

[13] FTC Staff Report: Self-regulatory Principles for Online Behavioral Advertising. www.ftc.gov/opa/2009/02/behavad.shtm.

[14] Google AdExchange. http://developers.google.com/ad-exchange/rtb/getting_started.

[15] Google Chrome Store AdBlock Plus Statistics. https://chrome.google.com/webstore/detail/adblock-plus/cfhdojbkjhnklbpkdaibdccddilifddb.

[16] libadblockplus: A C++ Library Offering the Core Functionality of Adblock Plus. https://github.com/adblockplus/libadblockplus.

[17] Most Popular Extensions :: Add-ons for Firefox. ttps://addons.mozilla.org/en-us/firefox/extensions/?sort=users.

[18] Mozilla AdBlock Plus Statistics. https://addons.mozilla.org/en-US/firefox/addon/adblock-plus/statistics/usage/?last=30.

[19] Publishers Watch Closely as Adoption of Ad Blocking Tech Grows. http://adage.com/article/digital/adoption-ad-blocking-tech-grows/297101/.

[20] Selenium: Web Browser Automation. seleniumhq.org.

[21] Writing Adblock Plus Filters. https://adblockplus.org/en/filters.

[22] Abdurhnam Albasir, Kshirasagar Naik, Bernard Plourde, and Nishith Goel. Experimental Study of Energy and Bandwidth Costs of Web Advertisements on Smartphones. In *IEEE MobiCASE*, 2014.

[23] Sebastian Angel and Michael Walfish. Verifiable Auctions for Online Ad Exchanges. In *ACM SIGCOMM*, 2013.

[24] Rebecca Balebako, Pedro Leon, Richard Shay, Blase Ur, Yang Wang, and L. Cranor. Measuring the Effectiveness of Privacy Tools for Limiting Behavioral Advertising. In *IEEE W2SP*, 2012.

[25] Paul Barford, Igor Canadi, Darja Krushevskaja, Qiang Ma, and S. Muthukrishnan. Adscape: Harvesting and Analyzing Online Display Ads. In *WWW*, 2014.

[26] Theodore Book and Dan S Wallach. An Empirical Study of Mobile Ad Targeting. *arXiv 1502.06577*, 2015.

[27] Michael Butkiewicz, Harsha V. Madhyastha, and Vyas Sekar. Understanding Website Complexity: Measurements, Metrics, and Implications. In *ACM IMC*, 2011.

[28] Xiaomeng Chen, Abhilash Jindal, and Y. Charlie Hu. How Much Energy Can We Save From Prefetching Ads?: Energy Drain Analysis of Top 100 Apps. In *ACM HotPower*, 2013.

[29] S. D'Ambrosio, S. De Pasquale, G. Iannone, D. Malandrino, A. Negro, G. Patimo, A. Petta, V. Scarano, L. Serra, and R. Spinelli. Mobile Phone Batteries Draining: Is Green Web Browsing the Solution? In *IEEE IGCC*, 2014.

[30] Amit Datta, Michael Carl Tschantz, and Anupam Datta. Automated Experiments on Ad Privacy Settings: A Tale of Opacity, Choice, and Discrimination. *IEEE PETS*, 2014.

[31] Ayman Farahat and Michael C Bailey. How Effective is Targeted Advertising? In *WWW*, 2012.

[32] Phillipa Gill, Vijay Erramilli, Augustin Chaintreau, Balachander Krishnamurthy, Konstantina Papagiannaki, and Pablo Rodríguez. Follow the Money: Understanding Economics of Online Aggregation and Advertising. In *ACM IMC*, 2013.

[33] David Gugelmann, Markus Happe, Bernhard Ager, and Vincent Lenders. An Automated Approach for Complementing Ad Blockers Blacklists. In *IEEE PETS*, Jul 2015.

[34] Saikat Guha, Bin Cheng, and Paul Francis. Challenges in Measuring Online Advertising Systems. In *ACM IMC*, 2010.

[35] Saikat Guha, Bin Cheng, and Paul Francis. Privad: Practical Privacy in Online Advertising. In *USENIX NSDI*, 2011.

[36] Hamed Haddadi, Pan Hui, Tristan Henderson, and Ian Brown. Targeted Advertising on the Handset: Privacy and Security Challenges. In *Pervasive Advertising*, pages 119–137. 2011.

[37] Michaela Hardt and Suman Nath. Privacy-aware Personalization for Mobile Advertising. In *ACM CCS*, 2012.

[38] Sunghwan Ihm and Vivek S. Pai. Towards Understanding Modern Web Traffic. In *ACM IMC*, 2011.

[39] Georgios Kontaxis and Monica Chew. Tracking Protection in Firefox for Privacy and Performance. *arXiv 1506.04104*, 2015.

[40] Aleksandra Korolova. Privacy Violations Using Microtargeted Ads: A Case Study. In *IEEE PADM*, 2010.

[41] Balachander Krishnamurthy and Craig Wills. Privacy Diffusion on the Web: A Longitudinal Perspective. In *WWW*, 2009.

[42] Balachander Krishnamurthy and Craig E. Wills. Cat and Mouse: Content Delivery Tradeoffs in Web Access. In *WWW*, 2006.

[43] Zhou Li, Kehuan Zhang, Yinglian Xie, Fang Yu, and XiaoFeng Wang. Knowing Your Enemy: Understanding and Detecting Malicious Web Advertising. In *ACM CCS*, 2012.

[44] Gregor Maier, Anja Feldmann, Vern Paxson, and Mark Allman. On Dominant Characteristics of Residential Broadband Internet Traffic. In *ACM IMC*, 2009.

[45] Gregor Maier, Fabian Schneider, and Anja Feldmann. NAT Usage in Residential Broadband Networks. In *PAM*, 2011.

[46] Hassan Metwalley, Stefano Traverso, Marco Mellia, Stanislav Miskovic, and Mario Baldi. The Online Tracking Horde: A View From Passive Measurements. In *Traffic Monitoring and Analysis*, 2015.

[47] Suman Nath. MAdScope: Characterizing Mobile In-App Targeted Ads. In *ACM MobiSys*, 2015.

[48] Vern Paxson. Bro: A System for Detecting Network Intruders in Real-Time. *Computer Networks*, 31(23-24):2435–2463, 1999.

[49] Enric Pujol, Philipp Richter, Balakrishnan Chandrasekaran, Georgios Smaragdakis, Anja Feldmann, Bruce Maggs, and K. C. Ng. Back-Office Web traffic on the Internet. In *ACM IMC*, 2014.

[50] Kent Rasmussen, Alex Wilson, and Abram Hindle. Green Mining: Energy Consumption of Advertisement Blocking Methods. In *ACM GREENS*, 2014.

[51] Philipp Richter, Nikolaos Chatzis, Georgios Smaragdakis, Anja Feldmann, and Walter Willinger. Distilling the Internet's Application Mix from Packet-Sampled Traffic. In *PAM*, 2015.

[52] Fabian Schneider, Bernhard Ager, Gregor Maier, Anja Feldmann, and Steve Uhlig. Pitfalls in HTTP Traffic Measurements and Analysis. In *PAM*, 2012.

[53] Kevin Springborn and Paul Barford. Impression Fraud in Online Advertising via Pay-Per-View Networks. In *USENIX Security*, 2013.

[54] Brett Stone-Gross, Ryan Stevens, Apostolis Zarras, Richard Kemmerer, Chris Kruegel, and Giovanni Vigna. Understanding Fraudulent Activities in Online Ad Exchanges. In *ACM IMC*, 2011.

[55] Narseo Vallina-Rodríguez, Jay Shah, Alessandro Finamore, Yan Grunenberger, Konstantina Papagiannaki, Hamed Haddadi, and Jon Crowcroft. Breaking for Commercials: Characterizing Mobile Advertising. In *ACM IMC*, 2012.

[56] Guowu Xie, Marios Iliofotou, Thomas Karagiannis, Michalis Faloutsos, and Yaohui Jin. Resurf: Reconstructing Web-Surfing Activity from Network Traffic. In *IEEE IFIP*, 2013.

[57] Xinyu Xing, Wei Meng, Byoungyoung Lee, Udi Weinsberg, Anmol Sheth, Roberto Perdisci, and Wenke Lee. Understanding Malvertising Through Ad-Injecting Browser Extensions. In *WWW*, 2015.

[58] Shuai Yuan, Jun Wang, and Xiaoxue Zhao. Real-Time Bidding for Online Advertising: Measurement and Analysis. In *ACM ADKDD*, 2013.

[59] Apostolis Zarras, Alexandros Kapravelos, Gianluca Stringhini, Thorsten Holz, Christopher Kruegel, and Giovanni Vigna. The Dark Alleys of Madison Avenue: Understanding Malicious Advertisements. In *ACM IMC*, 2014.