Cat and Mouse: Content Delivery Tradeoffs in Web Access

Balachander Krishnamurthy
AT&T Labs–Research
Florham Park, NJ, USA
bala@research.att.com

Craig E. Wills
Worcester Polytechnic Institute
Worcester, MA USA
cew@cs.wpi.edu

ABSTRACT

Web pages include extraneous material that may be viewed as undesirable by a user. Increasingly many Web sites also require users to register to access either all or portions of the site. Such tension between content owners and users has resulted in a "cat and mouse" game between content provided and how users access it.

We carried out a measurement-based study to understand the nature of extraneous content and its impact on performance as perceived by users. We characterize how this content is distributed and the effectiveness of blocking mechanisms to stop it as well as countermeasures taken by content owners to negate such mechanisms. We also examine sites that require some form of registration to control access and the attempts made to circumvent it.

Results from our study show that extraneous content exists on a majority of popular pages and that a 25-30% reduction in downloaded objects and bytes with corresponding latency reduction can be attained by blocking such content. The top ten advertisement delivering companies delivered 40% of all URLs matched as ads in our study. Both the server name and the remainder of the URL are important in matching a URL as an ad. A majority of popular sites require some form of registration and for such sites users can obtain an account from a shared public database. We discuss future measures and countermeasures on the part of each side.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Network Protocols—*applications*

General Terms

Measurement, Performance

Keywords

Content Blocking, Web Registration, Anonymity, Privacy

1. INTRODUCTION

Since the inception of the Web there have been numerous evolutionary steps in the basic exchange between content creators and users downloading content via browsers. The basic notion of a Web page has evolved from simple HTML (the language component) on a page to a complex set of objects of numerous types. The HTTP protocol component has undergone significant improvements and the last component—naming, i.e., URI, has been standardized. The business and social expectations enforced via forms to gather demographic and other user-related information, along with cookies and various other user-specific mechanisms have expanded greatly. Several popular Web sites have a significant component of extraneous material. This material includes advertisements, user-related data gathering mechanisms and flash animations. Such non-essential parts of the page sought by downloaders can be viewed as *extraneous* content from the perspective of the user and thus potentially undesirable. However, from the content owner's viewpoint, such material is integral for delivering content of interest to subsets of users. Advertisements are often linked to a click-through data-acquisition mechanism, which is a typical way to extract audience-related information and help generate revenue for the content owning Web site.

Increasingly many Web sites also require users to register before allowing them to access either all or portions of the site. Registration has always been necessary for content that requires payment, such as premium content of certain news sites or material available only to subscribers (e.g., New York Times' TimesSelect section, purchasing airplane tickets via sites such as Orbitz, etc.). Many sites have begun to require registration even for free material. Registration allows user to personalize the content they receive, but the registration information is used to augment demographic information about viewers of a site, increase ad revenues, and tailor advertisements. In the case of commercial sites registration data often includes the information about some form of payment (e.g., credit cards). Unlike credit card information, data gathered by non-commercial sites does not necessarily have to be verified for accuracy. It is not surprising that some of the information provided to such sites by a fraction of the users is inaccurate. Increasingly, many sites require users to supply an email address as well.

Popular sites on the Web routinely include extraneous content. Part of the extraneous content may be small images or specific script routines used to detect subsequent click access. Even without registration many Web sites gather information about users via traditional mechanisms such as cookies, tracking navigation patterns via the `Referer` HTTP header, or through third party sites such as DoubleClick.

Thus there is a basic tension between content owners and users that results in a "cat and mouse" game for what own-

WWW 2006, May 23–26, 2006, Edinburgh, Scotland.
ACM 1-59593-323-9/06/0005.

ers do in providing content and how users react in accessing it. Some users prefer to browse sites without having to register either due to concerns about privacy or to avoid the hassle of filling out forms. Users may prefer to receive content devoid of advertisements. Note that only *some* of the content downloaded as part of the normal downloading may be viewed as extraneous and such a view may be shared only by a subset of users. Many Web sites do deliver the requested content with no additional material. Such content may further be delivered without requiring any information from the user in the form of registration, cookies, or other privacy-related information.

Modern browsers, such as Firefox, have popularized mechanisms to block extraneous content indicating both the increasing prevalence of such material and the desire on the part of users to avoid such content. Among the numerous benefits touted are aesthetic improvement in viewing the page, reduced latency in obtaining desired portion of a page, absence of diversions such as advertisements, and avoidance of user tracking. Mechanisms also exist to bypass Web registrations.

Our work makes the following contributions:

1. We carry out a measurement-based study to understand the nature of the extraneous content that owners provide and its impact on performance as perceived by users.
2. We characterize how advertisements are distributed, which gives us an indication on how successful the blocking mechanisms are likely to be in the long run and the countermeasures that could be taken by the content owners to reduce or prevent such mechanisms.
3. We examine sites that require some form of registration to control access and the attempts made to circumvent it.

We look at these issues from a balanced perspective of both user and provider by not making a value judgment on extraneous content.

Sites have been created complete with a database of a large number of contributed user IDs and passwords that anyone could use to access many thousands of Web sites. Popular browser extensions enable seamless access to such databases and allow any user to access a Web site that would otherwise require registration.

The motivations common to avoiding undesirable content and registration are time and privacy. Registration may be a one-time hassle, but undesirable content may add to latency to each download. The joint examination of access enabling mechanisms, blocking mechanisms and their impact on privacy is to highlight the emerging nexus in the battle between the content owners and users. This tension is reflected in certain categories of Web sites more so than others. As one example, consider the growth of blogs [5, 10]. Several popular blogs have significant dependence on advertisements. Mechanisms that lead to reduced access of advertisements could prove problematic to such sites. Beyond blogs, numerous popular news and sports Web sites accrue revenue from users who click on advertisements. Secondary revenue generation by tracking those who follow advertisements is a key part of the business models of companies like Overture, Google etc.

The rest of the paper is organized as follows: Section 2 describes the general methodology and approach we take regarding issues with undesirable content and registration. Section 3 describes the details of the study we undertook to examine undesirable content and registration avoidance with the results of the study presented in Section 4. Section 5 discusses the manner in which content owners can and do react to the various blocking and avoidance mechanisms that are currently in vogue and what they might evolve into in the future. We conclude with a look at related work, a summary and future work.

2. METHODOLOGY AND APPROACH

We have framed the issues broadly in terms of content that is delivered by content owners and possibly filtered by users. In the reverse direction, users are reluctant to give the personal and/or demographic information sought by content owners resorting instead to extracting accounts from shared databases. The delivery of extraneous content and unwanted registrations on the Web highlight the tensions between the desires of users and of content providers.

The first issue is defining extraneous content: a subjective issue dependent on individual users or groups of users. There is a consensus on what *kinds* of content may generally be viewed as extraneous and thus undesirable from the viewpoint of latency. Advertisements and resources that are placed on a user's site solely for the purpose of gathering user-specific data, i.e., content extraneous to the core content of the page, could be considered undesirable. Pop-up and pop-under windows are annoyances that modern browsers often suppress.

The second issue is identifying syntactically the set of resources that constitute such content, examining common features, and being able to describe them. We can list all the URIs that are viewed as extraneous and block their downloading or rendering by the browser. This is tedious, exhaustive, and impractical. Another extreme technique is setting browser preferences to turn off downloading of all images and any script language entities. But some images may be part of a news story that is of considerable interest to the user and some scripts are necessary simply to display a page properly. Patterns or regular expressions that can describe a large number of undesirable URIs makes for a terse specification that can be used to block the displaying and/or downloading of such content. This intermediate popular technique behooves studying.

The third issue is the benefits of such content and registration to content owners versus the perceived downside to the users. For content owners, the ability to deliver ads may be crucial for revenue generation purposes as is demographic information gathering via registration. The impact in terms of performance, hassles in accessing a site, and loss of privacy are what concern users.

The fourth issue is *where* blocking and measures against blocking are most effective. From the user's perspective if undesirable content is detected and automatically bypassed in downloading they can benefit the most. Likewise, bypassing registration by using a local database of userids for popular sites can speed up access. For content owners, ensuring that their ad URIs are not matchable against publicly distributed patterns and their registration userids are not present in public databases requires additional work. They can resort to dynamic URIs and outsource their generation, and blacklist publicly available userids. Note that the interesting question of what fraction of users currently block ads is not studied here and left for future work.

3. STUDY

The tension between content owners and users has led to tools that provide content blocking techniques and allow users to bypass requested registrations. We carried out a study by examining questions in four categories:
Characterizing presence and delivery of ad content:

- What percentage of popular sites and pages on the Web contain objects deemed extraneous by some users?
- Do these percentages change with the types of sites?
- What is the proportion of ad content in pages?
- Do content owners serve ads or do they outsource it to external ad companies? What is the distribution of ads amongst the servers/companies?

Performance impact on users:

- What is the impact of ads on page download performance?
- How is delivery of ad content impacted as a result of users enabling or disabling Java/JavaScript/images?

Detection of ad content and measures by content owners:

- How has the detection and blocking of ads changed over time with syntactic techniques?
- To what extent are ads detected from the server name that serves the content vs. rest of the URL?
- How well can the semantics of extraneous content be matched? If syntactic matching is defeated by content owners, would semantic matching (e.g., checksums of extraneous content) help in blocking?

Registration requirement and avoidance:

- To what extent do sites use registrations? Does this change across different types of sites and pages?
- What is the correlation of these sites with those sites containing ads?
- How effective are the tools in avoiding registration?

3.1 Web Sites for Study

As a broad representation of sites, we selected a list of around 1300 URLs from Alexa's popular sites in the English language [2]. Alexa is a popular historical Web archiving site that has long monitored popularity of Web sites in various categories. Alexa breaks sites into categories, necessary to answer one of our study questions; hence our use of it as opposed to other sources used in our earlier work [8].

We selected 12 categories: arts, business, computers, games, health, home, news, recreation, reference, regional, science and shopping. We retrieved the top 100 sites for each of these categories. We added a 13th category—political, from a published list of the 100 popular political sites [14], as many of these sites are exclusively funded by ads.

A summary of the 13 categories broken down by DNS Top-Level Domain (TLD) is shown in Table 1. The table shows that 988 (76%) are from the .com domain, which is the dominant domain for all but two categories. Other domains contribute non-trivial numbers to some categories, but do not contribute significantly overall.

Table 1: Category Breakdown by Top Level Domain

Category	com	org	gov	net	edu	other	Total
arts	82	3	0	5	0	10	100
business	93	0	1	2	0	4	100
computers	86	6	0	5	0	3	100
games	96	1	0	3	0	0	100
health	51	20	15	3	8	3	100
home	87	4	7	1	0	1	100
news	81	4	1	2	0	12	100
political	76	18	0	5	0	1	100
recreation	85	2	4	2	0	7	100
reference	45	5	6	0	28	15	99
regional	67	1	4	3	0	25	100
science	42	9	30	0	7	12	100
shopping	97	0	0	0	0	3	100
Total	988	73	68	31	43	96	1299
global 500	315	4	4	34	0	143	500

Some sites are listed in multiple Alexa categories. For example `www.cnn.com/` is both an art and a news site. Filtering duplicates we end up with 1158 unique sites. The Alexa source also lists multiple "sites" from the same server. For example, `www.cnn.com/tech/` is a site listed in the science category. Because the Alexa list contains multiple entries from the same Web site, we henceforth use the term "page" to refer to an entry so the data set contains 1158 unique pages with 1116 unique server names.

These are the primary list of pages studied and reported on in this work. As a means to include a wider variety of popular pages, we also obtained the Alexa "Global Top 500" list of 500 globally popular pages, which include both English and non-English pages. Results using this data set are reported as appropriate in the results. A summary of this list based on TLD is shown as the last line in Table 1 for comparison. It indicates many more pages from different countries. Manual inspection of .com pages indicates some of these are non-English pages. 180 of the pages in the global 500 list are also in our primary Alexa list.

3.2 Data Gathering

To answer questions about the proportion of ad content for Web pages in our study set, we configured a Firefox browser to send requests to a non-caching proxy (co-located on the same machine as the browser) that logs request URLs made by the browser, the size and response code returned in response. We created a new instance of Firefox for each page and closed it after 45 seconds—long enough for the page to load. The browser is configured to execute JavaScript code and Java applets, download embedded images, prevent popups and the cache size is set to zero for the tests.[1] The simple proxy automatically handles redirections returned by a server, but does not handle HTTPS and thus such requests are not successfully retrieved.

We used this browser-based retrieval approach over the easier-to-automate script approach we had used in previous work [8] because of concerns on the potential impact of JavaScript to the set of objects requested by a browser.

[1]Firefox's back-forward cache (http://developer.mozilla.org/en/docs/Using_Firefox_1.5_caching) did not affect our study as we used an earlier Firefox version–1.0.7.

3.3 Adblock

We chose Adblock [1], one of the most popular Firefox extension (https://addons.mozilla.org/extensions), as the syntactic content blocking tool to use in our study. Adblock works by allowing a user to specify a set of pattern rules, each of which can either be a literal match along with the wildcard '*' or can be a full regular expression. The URLs of all objects that must be retrieved by a browser are compared against these rules and if a match occurs then the object is not retrieved at all. One version of Adblock allows a preference for retrieving but not rendering the ads.

While users have the capability to define their own set of rules, the work of a user "G" has led to a set of rules dating from July 2004 that incorporates input from others [6]. This ruleset, known as "Filterset.G", is commonly accepted as best practice for using Adblock to block extraneous content and has led to development of a separate Firefox extension to automatically incorporate new Filterset.G rules into a user's set of Adblock rules. We use the Filterset.G ruleset to define what is extraneous content from a list of URLs retrieved for a page and use the 2005-10-10 version of the rules for all results in our study. We note that there have been concerns raised regarding applicability of Filterset.G for non-American Web sites. We also examine the impact of using other rulesets as well as "whitelists" as one part of our study.

For our analysis, Filterset.G rules are converted to Perl regular expressions (Perl and JavaScript use the same regular expression syntax). This approach allows us to apply different filtersets against a set of URLs regardless of how the URLs are obtained.

3.4 BugMeNot

BugMeNot [4], created in August 2003 by an Australian who seeks to remain anonymous, has received considerable attention in the popular press [7, 11]. This evocatively named site contains a database of accounts contributed by users that can be used by others to bypass Web registration. The number of sites that have been 'liberated' from registration nearly doubled between March and October 2005 from 57,189 to 93,550. The number of unique visitors per day has more than tripled in that period from below 15,000 to around 50,000. Over 20,000 accounts exist for the 100 most requested Web sites on BugMeNot. It is important to note that these numbers are from the creator of BugMeNot, obtained via personal communications either with members of the press (in March 2005) or with one author of this paper (in late October 2005). While we have no reasons to doubt these numbers, verifying them is hard and requires a distributed reverse engineering attempt.

The site allows two approaches for querying the database. First, the site itself has a Web form in which the URL of a site can be entered and a userid and password retrieved, if available. Second, the database can be accessed using a HTTP GET request with the desired URL as part of the request. This latter approach is used by a BugMeNot Firefox extension and is the method used in our study, although we use it directly rather than as a Firefox extension. Note that an Internet Explorer plugin for BugMeNot can be found at http://www.unixdaemon.net/zips/bugmenot.zip.

This approach allows us to answer the question of whether or not a BugMeNot account exists for a site, but not the number of accounts available in BugMeNot for a given site. This question requires measurements at different times from various locations, which we also did as part of this study. We also verified that for the popular news Web site `www.nytimes.com` there are at least a hundred distinct accounts available by sending sequential requests indicating supposed failure of an earlier account returned from BugMeNot. We did not actually test the success of these accounts with the Web site.

4. RESULTS

Based on the methodology described in the previous section and resulting data collection, we now present results of various tests to answer questions posed about the impact of extraneous content and avoiding registration.

4.1 Ad Results by Category

Using the retrieval method described in Section 3.2, the pages for all categories in the Alexa data set were retrieved at different times over a period of a few days. Table 2 shows the results derived from the retrieved data. The first entry in the table shows that 1113 pages were successfully retrieved and of those pages, 622 (56%) contained at least one object matched using the Filterset.G ruleset. These results exclude 10 pages for which the container page itself is matched as an ad, which indicates that the page was incorrectly identified as an ad or this is a page that serves ads. A separate check of the 10 pages by hand indicates the latter. Unsuccessful retrievals were primarily due to redirections to a URL via HTTPS, which failed at the proxy.

Table 2: Ad Content Presence by Category and TLD

Category/ TLD	Pages with Ads/Total (%)	Category/ TLD	Pages with Ads/Total (%)
all	622/1113 (56)	reference	32/99 (32)
arts	76/98 (78)	regional	59/98 (60)
business	61/98 (62)	science	31/95 (33)
computers	47/88 (53)	shopping	54/97 (56)
games	61/95 (64)	com	532/832 (64)
health	40/99 (40)	org	24/65 (37)
home	60/97 (62)	gov	3/50 (6)
news	83/98 (85)	net	13/27 (48)
political	61/88 (69)	edu	5/43 (12)
recreation	46/95 (48)	global500	266/457 (58)

Subsequent results in Table 2 show that the percentage of pages within a category containing ads ranges from 32% for reference to 85% for news. The next group in Table 2 shows the results based on the TLD of the container page. As shown in Table 1, the .com domain accounts for most of the pages in the study set and not surprisingly, these pages have the highest incidence of ads. The last entry in Table 2 shows results for the global 500 list. Of the 457 pages with successful retrievals, 149 of them are the same as our primary dataset. The global 500 pages show just a bit higher percentage of pages that contain ads.

Table 3 shows statistics for the 622 pages in Table 2 that contain at least one matched ad. Examining the number of different servers involved in delivering a page, on average, a majority of them actually provide ad objects. From a performance standpoint, this result implies additional DNS lookups and reduced opportunities for aggregation of object retrievals over persistent HTTP connections. The remaining results for these pages show that on average ads account for about one-sixth of objects and bytes.

Table 3: Page Statistics by Category and TLD

Category/ TLD	Avg. Ads/Total for Pages *with* Ads		
	Servers	Objects	KBytes
all	3.2/5.7	8.1/52.0	48/295
arts	4.1/6.5	8.7/57.7	58/392
business	3.0/5.6	10.2/52.3	48/305
computers	2.9/5.6	6.7/46.0	32/206
games	2.4/5.1	4.9/54.6	31/401
health	3.5/5.3	8.4/45.4	63/270
home	3.5/6.2	8.1/50.2	44/250
news	4.6/7.8	13.3/61.7	80/347
political	3.5/6.3	10.8/43.6	94/314
recreation	2.7/4.7	5.0/55.7	28/291
reference	3.5/5.2	7.1/36.9	42/208
regional	2.6/5.5	9.7/54.6	41/259
science	3.1/4.6	7.0/46.9	52/256
shopping	2.0/4.7	4.4/56.5	20/248
com	3.3/5.9	8.5/52.7	51/301
org	2.2/3.7	6.4/44.8	49/307
gov	1.0/2.0	1.0/29.7	31/234
net	3.8/6.7	7.2/40.0	35/259
edu	1.0/1.4	2.2/48.0	9/184
global500	2.8/5.9	8.0/56.9	40/285

Per-category statistics show that for most categories the majority of servers used for serving content do so for ad objects. Political pages containing ads show the highest concentration of ads on average with 25% of the objects and 30% of the bytes on these pages due to ads. Statistics for the TLDs in Table 3 show the highest concentration of ads on a page for the .com and .net domains. Results for the global 500 pages demonstrate consistency with our primary results over a global dataset.

4.2 Ad Results by Retrieval Methodology

In the next phase of our study we measured the presence of ad content under three retrieval configurations of the Firefox browser to examine the impact of these configurations on the amount of total and ad content that is retrieved. In all cases, Adblock is disabled.

1. normal: images and JavaScript/Java on
2. nojj: images on, JavaScript/Java off
3. noimgjj: images off, JavaScript/Java off

The first method is the normal approach used for all other retrievals in our study; the other two examine effect of disabling Java/JavaScript interpretation (disabled together despite being separate mechanisms) and image downloading. These three methods were used to retrieve the objects for the 100 pages in the news category of Table 1. All retrievals were made during the same weekday afternoon. All 100 pages were retrieved with one Firefox configuration before the next configuration was used. Results for all three methods across 99 pages successfully retrieved are shown in Table 4.

Averages for servers, objects and bytes along with median values are shown in Table 4. While not shown, relative differences in the distributions of servers, objects, bytes and download time are consistent with the results shown in the table. The results for all objects are significant in two ways. First, for the normal retrieval method an average page contains 48 objects retrieved from 7.7 servers. In previous work [8], we observed a relatively small number of pages where multiple servers were used to serve the contents of a page. Clearly this approach is now widely used. Second, turning off the interpretation of Java and JavaScript code has a relatively small effect on the number of objects retrieved, but a much larger effect on the number of servers and total bytes, as well as the measured (at the proxy) total page download time. Note, because alternate HTML content may be interpreted if scripting is turned off for a page, the objects retrieved using the "nojj" method are not guaranteed to be a subset of the objects retrieved using the "normal" CEW: rewrote the following a bit method. These results indicate that Java/JavaScript has a significant effect on performance measures of a page and given that we observed no `.class` objects retrieved in this portion of the study, the effect is entirely due to JavaScript. Turning off images in addition, not surprisingly, has an even larger impact on objects and total bytes, and an equal impact on download time as turning off JavaScript.

Table 4: Page Statistics for Retrieval Methods

Retrieval Method	Average Number Ads/Total			Median Download Time
	Servers	Objects	KBytes	
normal	4.4/7.7	12.7/48.0	84/341	10.4s
nojj	1.4/4.1	4.8/42.3	24/226	7.6s
noimgjj	0.3/1.5	0.5/3.2	1/87	1.3s

We also applied the Filterset.G filter set to the objects retrieved for results in Table 4 to obtain the subset of these objects matched as extraneous content in the form of ads. Results for this test are also shown in Table 4. The most striking result concerning ads is the sharp drop-off (12.7 to 4.8, 62%) in matched ads between using the normal retrieval method and turning off Java/JavaScript execution. This drop-off is much higher than the relative drop (48.0 to 42.3, 12%) for all objects. This result indicates that content providers are using the execution of Java and JavaScript as a means to retrieve content marked as ads. Simply turning off the execution of this code in the browser results in a significant decrease in the amount of ad content that is downloaded.

4.3 Performance Impact of Ad Content

The next portion of our study was directed at answering the question on how the presence of ads on a Web page impacts the download performance. We configured our Firefox browser in normal mode (Image/Java/JavaScript enabled) and directed to our recording proxy. We installed the Adblock extension with the rules of the Filterset.G 2005-10-10 rules, and configured it to not download any object matching a rule. Via the Firefox profile option we created two identical profiles—one with ad blocking enabled and one disabled.

Using these two profiles, we set up an experiment where the pages in the Alexa news category were used as input. An instance of the Adblock-enabled browser was created for the first URL in the news list. After 45 seconds this instance was killed and an instance of the Adblock-disabled browser was created for the first URL in the list. After 45 seconds this instance was killed and this pattern repeated for all pages in the news category list. This methodology was used to minimize differences in the page content between the two

Table 5: Page Statistics and Download Times for Adblock Disabled and Enabled

Timeframe (all EST)	Page Count	Adblock Status	Avg. Ads/Total for Pages (% Reduction)			Median Download Time	Latency Reduction
			Servers	Objects	KBytes		
weekday midday	86	disabled	5.2/8.9	14.8/66.3	73.2/339.7	12.7s	
	86	enabled	-/3.8 (57)	-/48.3 (27)	-/248.7 (27)	8.6s	32%
weekday afternoon	83	disabled	5.3/8.9	14.7/66.6	78.5/337.1	14.2s	
	83	enabled	-/3.8 (57)	-/49.1 (26)	-/242.5 (28)	9.2s	35%
weekend morning	86	disabled	5.3/9.0	14.6/65.3	75.7/336.2	10.3s	
	86	enabled	-/3.7 (59)	-/48.5 (26)	-/246.0 (27)	7.0s	32%

retrievals while performing the two downloads under similar network conditions.

We repeated the experiment thrice: during midday and late afternoon of a weekday, and in the morning of a weekend day. All statistics, including the download time (from first object request to last object retrieval), were recorded at the proxy. Results for the average number of total servers, objects and Kbytes needed to render each page are shown in Table 5 for Adblock disabled and enabled. The counts for ad objects are also shown for the Adblock-disabled experiments when Adblock does not block these objects. Although not shown, differences between the distributions of each pair of results is comparable to the averages. Results are only shown for pages that do contain ads when Adblock is disabled. The results are comparable to those shown for the news category in Table 3.

The midday results show a 57% reduction in the number of servers accessed when Adblock is enabled. There is a 27% reduction both in number of objects retrieved and number of bytes downloaded with Adblock enabled. The other timeframes show similar results for the reduction in servers, objects and bytes.

The last column of Table 5 shows how these differences translate into median download latency reductions. We see a 33% median reduction across the three time periods—consistent with both mean (27-37%) and 90th percentile (28-46%) figures (not shown in Table 5). The latency results are expected given the significant reduction in the number of servers contacted and the number of objects and bytes downloaded.

Observation of request times recorded at the proxy indicates that Firefox makes up to four parallel requests at a time to the proxy. Persistent connections are not used between the browser and proxy, or between the proxy and a server. We believe the timing difference between the Adblock-enabled and disabled experiments is due to the obvious difference of objects and bytes between the two experiments and the significantly larger number of DNS requests that must be made—despite DNS caching effects due to the Adblock-disabled retrieval being made after the Adblock-enabled retrieval. These additional DNS lookups are likely to return non-cached results as many ad servers employ CDNs, which return low time-to-live values. While latency differences will vary from different client sites, the differences observed in these experiments indicate that the removal of ads from the set of downloaded objects for a page does yield a significant reduction in the observed latency.

4.4 Ad Server Characterization

Table 6 shows the distribution of servers serving advertisements in our study of popular Web sites based on Alexa's ranking. We first identified servers involved in delivering advertisements and then we merged them to the second-level domain name since many companies operate internationally and some have tailored domain names for their customers. The top 10 ad-delivering-companies (computed by merging to 2nd-level domain name) for 5006 unique ad URLs found in pages of our primary Alexa data set are responsible for roughly 40% of all ads, the top 20 for about 50%, and the top 50 for over two-thirds of ads. Using the same methodology for 2667 unique ad URLs obtained from pages in the Alexa Global 500 list we obtained similar results with the top 10 accounting for roughly 35%, the top 20 for roughly half and the top 50 for over 70%.

Table 6: Top 10 Ad Servers for Alexa Data Set

Ad Server	Proportion (%)
doubleclick.net	8.8
advertising.com	5.2
falkag.net	4.7
atdmt.com	4.4
blogads.com	3.7
akamai.net	3.2
zedo.com	3.1
2mdn.net	2.8
com.com	1.9
2o7.net	1.9

We then took a closer look at the server portion of the advertisement URLs and looked up the corresponding IP addresses. We were interested in seeing if the advertisement companies and/or content owners were resolving the names to different IP addresses based on suspected geographic location of the clients. There were a total of 872 unique servers of which 305 reported aliases in *nslookup*. We then ran *dig* from 53 different PlanetLab [13] nodes to look up each of these advertisement servers and logged the number of unique IP addresses returned for each. 103 servers, or about a third of the servers, returned a single IP address in all lookups and these do not do any DNS-based load balancing. Two servers belonging to one large online retailer returned between 97 and 98 different IP addresses indicating that they were using load balancing DNS servers such as Akamai or Speedera. Not all servers used an external server—some did their own load balancing (e.g., freeonlinegames.com).

The interest in load-balancing DNS stems from the fact that these companies can use dynamic URLs belonging to their load balancing servers that cannot be safely blocked. If an advertisement server uses the name of a CDN (content distribution network), the server portion cannot be safely blocked as the CDN might serve desirable content. This naturally would force blockers to examine the non-server portion of the URI. Anti-blocking measures by content owners and advertisement servers would then focus on the non-server URI portion.

4.5 Filterset Analysis

We also examined the influence of the filterset used on the results we obtained focusing on three questions: 1) the contribution of different rules in the primary filterset we used; 2) the impact of whitelists, a feature present in the improved version of Adblock called Adblock Plus and recently added to Adblock), on the results; and 3) the stability of results using Filterset.G rules from various periods of time.

The Filterset.G version we used is labeled 2005-10-10a.txt and has 155 rules, 47 of which are true regular expressions containing special characters and the remainder are used to literally match specific server names. In our primary results, 65 (42%) of the rules matched at least one URL (some URLs were matched more than once) for a total of 6933 rule matches. The most-matched rule is a long regular expression with many alternations that resulted in 56% of the rule matches. The second most matched rule is another regular expression that resulted in another 11% of the rule matches. In all, the top five matching rules accounted for 80% of the rule matches and the top ten matching rules accounted for 89% of the rule matches. Two of the top five and four of the top ten were literal rules.

The whitelist feature allows users to specify rules for not blocking URLs that would otherwise be blacklisted. Like regular Adblock rules, the Filterset.G web site maintains whitelist rules; we downloaded wb-2005-10-10a.txt with 21 whitelist rules. For example, a rule exists to whitelist any URLs from the .edu, .gov or .mil domains or URLs with news.yahoo.com as substring. Applying these whitelist rules to the 5483 matched URLs in our study eliminated only 55 (1%) URLs—not a significant impact on out set of matched URLs.

Past versions of Filterset.G rules are available along with a log of changes made between versions [6]. We downloaded filtersets from five different time periods dating back to August 2004 when the first filterset was made available. We applied each of these filtersets to the entire set of URLs downloaded in our study. After the first time period, the results show relative stability as previous rulesets produce similar counts of matched URLs as well as the count of URLs that intersect with those from the 2005-10-10 ruleset.

4.6 Understanding Why Ad URLs are Matched

We next examined what component of a URL is matched by a rule to determine it is an ad. A content provider may want to understand what aspect of a URL causes it to be matched by a filterset rule.

We started with the 5483 URLs in our data set matched by at least one rule in the 2005-10-10 Filterset.G ruleset. Each URL is of the form "`http://`*server/path*" so we created two parallel data sets—one for all servers and one for all paths. We then applied the Filterset.G rules to each data set separately to determine if URLs are being matched because of the server they are from or the path they contain. The separate results were then recombined for each URL. Results from this analysis are shown in Table 7.

The first line of results in Table 7 shows that of 5483 matched URLs, 38% were matched by the server component alone, 48% were matched by the path component alone, 10% were matched by each component, and 5% were matched by neither component. The last group is interesting as it says the match is lost once the server and path are separated. We repeated the same analysis for the 5006 matched URLs that are unique with the results shown in the second line of table. The third line shows results for the 872 unique server names from the list of 5483 and the last line shows results for the 4939 unique paths in the data set. Results from all data sets are consistent.

Table 7: Component Contribution to Matched Rules

Data Set	Total Count	Match Server Alone	Match Path Alone	Match by Each	Match by Neither
Matched URLs	5483	38%	48%	10%	5%
Unique URLs	5006	39%	48%	8%	5%
Unique Servers	872	37%	-	-	-
Unique Paths	4939	-	56%	-	-

From the perspective of content providers, the results indicate that obfuscating the path component of the 5483 matched URLs to avoid matches can reduce the number of matches by 48%. Obfuscating the path component for the 38% of URLs matching the server component has no effect on the overall match nor is there any effect for the 10% where each component is matched because eliminating the path match still leaves the server match. The remaining 5% of URLs are matched because of both the server and path components so these URL matches would also be lost if the path was changed.

4.7 Semantic Analysis of Ad Content

We also examined matches in the content of different ad objects. We fetched all 5006 advertisement resources and computed their MD5 checksum. We found that 23% of the objects shared an MD5 with another object leaving 77% unique objects. The top 10 objects with identical MD5s accounted for around 15% of all objects, but there was a significant drop off after that—the top 50 objects accounted for just over 20% of all resources. The most common object was the single pixel (1x1) GIF image ('Web bug') used by various companies to place on different locations of a page to track a user's behavior. This image representing nearly 4% of all advertisement objects was found on 81 different sites in our Alexa data set. A variation on the 1x1 pixel used for similar purpose was found to occur 2.5% times overall on 43 different sites. Given the wide dispersion of MD5s it is not easy for blockers to use the MD5 route and content owners can simply tweak the image without perceptible changes and defeat MD5s. This is similar to what email spammers did to defeat spam-blocking based on content hashes by adding random strings in their spam email.

4.8 Breadth of BugMeNot Database

We next studied the issue of registration for pages on the Web along two axes using the BugMeNot database.

We examined the "breadth" by examining the presence of BugMeNot accounts across all categories in the Alexa data set. We queried the BugMeNot database for each of the sites containing pages in the our 13 categories. The results, shown in Table 8, indicate that of the 1113 pages included in the ad content analysis in Table 2, 648 (58%) of these pages contain at least one account in the BugMeNot database. This figure is slightly higher than the proportion containing ads in Table 2. A total of 439 (39%) pages contain ads and have a registration account while 831 (75%) contain at least one.

Table 8: BugMeNot Accounts by Category and TLD

Category/ TLD	Total Pages	Pages w/ Acct (%)	Pages w/ Ad & Acct (%)	Pages w/ Acct or Ad (%)
all	1113	648 (58)	439 (39)	831 (75)
arts	98	81 (83)	68 (69)	89 (91)
business	98	63 (64)	47 (48)	77 (79)
computers	88	66 (75)	38 (43)	75 (85)
games	95	78 (82)	53 (56)	86 (91)
health	99	32 (32)	21 (21)	51 (52)
home	97	55 (57)	40 (41)	75 (77)
news	98	88 (90)	77 (79)	94 (96)
political	88	34 (39)	32 (36)	63 (72)
recreation	95	48 (51)	23 (24)	71 (75)
reference	99	43 (43)	20 (20)	55 (56)
regional	98	63 (64)	45 (46)	77 (79)
science	95	38 (40)	21 (22)	48 (51)
shopping	97	59 (61)	33 (34)	80 (82)
com	832	547 (66)	381 (46)	698 (84)
org	65	18 (28)	12 (18)	30 (46)
gov	50	8 (16)	2 (4)	9 (18)
net	27	17 (63)	10 (37)	20 (74)
edu	43	10 (23)	1 (2)	14 (33)
global500	457	245 (54)	168 (37)	343 (75)

The next set of results in Table 8 show results on a per-category basis. They show a range in BugMeNot presence from 90% in the news category pages down to 32% for health category pages. In comparison to results in Table 2, the health and political categories show the biggest relative drop from ad presence to registration presence.

The next group in Table 8 show results based on TLD. Again the .com TLD shows the highest presence of BugMeNot accounts. Finally, we also queried the BugMeNot database for the pages in the global 500 and found that 54% of these pages have a BugMeNot account.

It is important to note that all of these results for pages with Web registration is a lower bound as the BugMeNot database is unlikely to contain accounts for all such pages in our data set. As a test of this hypothesis, we obtained the list of pages without a BugMeNot account for the arts, games and news categories. We manually inspected each of these pages with Firefox and looked for cases where registration is part of the page, but no BugMeNot account is available. We found some cases where pages contained logins requiring personal or financial information for which the lack of a BugMeNot is expected. We did find a total of six cases, two in each category, where a BugMeNot account could be available, but was not. On the other hand, these results indicate almost all pages that could have BugMeNot accounts in these categories do have them meaning the BugMeNot penetration is significant.

4.9 Depth of BugMeNot Database

Finally, we examined the "depth" of the BugMeNot database by analyzing how many accounts are available for each Web site. We also examined how it responds to requests from multiple clients at the same time—does it return the same account or does it cycle through multiple accounts stored in its database?

For analyzing the number of accounts for each server, we ran an experiment using 84 unique Web sites supplying pages for the news category known to have a BugMeNot account from Table 8. From a single client, we retrieved a BugMeNot account for these sites every 8 hours over a four day period for a total of 12 retrievals. In approximately 50% of the cases, the same account information was returned for each of the 12 retrievals. In approximately, 25% of the cases exactly two different accounts were observed amongst the 12 retrievals. The remainder of the cases returned more than two accounts with a maximum of 10 for `www.nytimes.com`.

Separately, as discussed in Section 3, we carried out a distributed reverse engineering attempt to examine if different accounts were returned to different users from the BugMeNot database. We generated simultaneous requests to the BugMeNot database from 50 PlanetLab nodes for a set of 61 Web sites identified as popular by Alexa, requiring registration and known to have a presence in the BugMeNot database. We found that 59 (97%) of the tested sites returned the same account information for each of the 50 nodes. Only two of the sites yielded a second distinct account during the test and none yielded more than two. The combined results of these experiments indicate that approximately half of the tested sites contain multiple accounts in the BugMeNot database, but the algorithm used to select an account has a strong tendency to return the same account over a short period of time. This tendency makes it easier for content providers to detect the use of these accounts by observing a large number of registrations with the same account over a short period of time.

5. DISCUSSION

Our results largely present current practice on the use of extraneous content and registrations on the Web as well as the effectiveness of mechanisms that users employ to avoid them. In this section, we examine how such mechanisms affect the content owners, what measures they can take to circumvent the defenses, and the expected future in this cat and mouse game.

The two key reasons for requiring registration or enhancing the requested content with extraneous content are the potential for earning revenue and ability to gather information about users. Extraneous content in the forms of advertisements might result in sales of products some of which may be directly related to the site (e.g., a popular news site advertising a sister publication). Tracking supplements data they might be already gathering using other techniques and the information could be sold. Other reasons include tailoring the content better to suit the demographics of the current readership or combining the data with other related sites to obtain broader readership information. The information can also be used to re-orient the content based on the duration of time spent in different sections of a Web site. The time taken to download a page, the inter-arrival rate of requests can be used to alter the content or the manner of delivery [9].

As a result of selective downloading on the part of users (e.g., via Adblock or disabling JavaScript) content owners will not know if advertisements are displayed or viewed. If extraneous content is never downloaded then the probability of ad-related revenue is dramatically reduced. The inability to gather demographic information could lead to further lowering of ad revenue as it would not be possible for sites to command high ad prices even if sites have a desirable readership demographic. In the absence of registration in-

formation they would not be able to correlate reading habits of subscribers and distinguish it from non-subscribers.

5.1 Measures and Countermeasures in Blocking URLs

There are several technical means by which content owners can blunt the content and registration avoidance techniques. The steps would inevitably result in countermeasures on the part of those who advocate blocking and avoidance schemes. The increasing popularity of Firefox, numerous contributed extensions, steady augmentation of patterns to match and block unwanted content with the added ability of automated periodic updates, and the widely perceived benefits of avoiding registration and maintaining a large number of user ids and passwords all point to an inevitable arms race.

One measure against blocking advertisements is to make them indistinguishable from regular content. For example, if a news article has in-line text advertisement or if a news-related image has a merged advertisement graphic then blocking would result in not receiving wanted content. The content owners could seek to dramatically raise the false positives. If a popular widely blocked pattern is `*/Realads/*` then interesting content could be stored in URLs that match this pattern forcing users to either remove the pattern from the block list or resort to whitelisting a large number of URLs. Such a technique may lead to long term success for content owners and advertisers against most blocking mechanisms. A countermeasure would have to process the HTML or image and automatically separate out the extraneous content—a non-trivial task.

The second measure is to adopt spammers' techniques. Just as most syntactic schemes to block spam have largely been circumvented by spammers, content owners and advertisement creators and distributors can periodically change the URL patterns to negate pattern-based blocking. Currently, per Table 6, a large fraction of advertisements are delivered by a few ad servers and the same ad servers deliver the same advertisements for multiple sites (Section 4.7). A container page, such as http://www.nytimes.com/index.html, is controlled by the New York Times, which assembles the contents of the page including all the URL strings referring to various advertisements. For frequently changing pages, such as most news sites, it would be an added burden to steadily change the URL strings while referring to the same advertisements. Also, rotation of a few URLs could still result in their being aggregated into patterns and blocked. In other words, each time a `index.html` is assembled the set of advertisements should have a different URL. Whether managed by the parent content owner site or an ad server site, this would require significant work—most of which is wasted on users who do not use blocking techniques. A mechanism depending on changing URL strings is thus a sub-optimal measure. However, as we pointed out earlier, the URL can be obscured by resorting to use of CDNs which can maintain a private and changing mapping of URLs to prevent syntax-based blocking without fear of false positives.

The third measure, one which is already prevalent on a few sites, forces readers to click past an advertisement to reach the desired content. This is relatively foolproof but can annoy many readers who may just not visit the site. Blocking this URL is counterproductive as the content would simply not be accessible. However new extensions can be written to automatically skip the advertisement by mimicking the action required by the user. Note that the Firefox extension mechanism is powerful and virtually any action required to be carried out by a user can be programmed into an extension and triggered upon the matching of a URL pattern.

5.2 Measures and Countermeasures in Registration Avoidance

Sites that require registration and receiving automated responses generated from a public database such as BugMeNot can take a variety of measures. Tracking popular userids and blocking them can reduce effectiveness of avoidance techniques. The random selection feature of BugMeNot and periodic rotation may increase complexity but is hardly foolproof. However, our study has shown that for requests coming in close temporal proximity the same userid and password combination is returned. A site could look for clients sending the same userid within a short time from different IP addresses to identify avoiders. By temporarily disabling an account the site can force even the 'legitimate' owner of the account to re-register with the site and reduce the risk of the account being submitted to a public database. Some sites (e.g., BeerAdvocate.com, http://beeradvocate.com/tos.php) have added clauses in their terms of service specifically mentioning registration bypassing sites and indicating that sharing of accounts is forbidden.

Next, the content owners can simply reverse engineer public databases, obtain a list of the stored userids, and blacklist them. BugMeNot could keep track of frequency of requests and block requests from suspected content owners but it is easy for content owners to distribute queries (as we did in our study). However, as a countermeasure BugMeNot can ensure that the database keeps growing. Insisting that their users register an account every so often (say once monthly) in order to use their service for all the sites in the database would result in constant augmentation. A BitTorrent like tit-for-tat mechanism would allow BugMeNot to enforce periodic new registration by its users and defeat blacklisting by content owners. The choice of site to create a new registration could be a function of the popularity of requests for userids for that site.

Other countermeasures that can be taken by BugMeNot are ensuring long rotation cycles between fetches, distributing their databases to ensure genuine randomness among the set of userids available for a given site. For example, there can be multiple databases and the user's query is routed to a random one and a random userid is then selected from that database. BugMeNot can also resort to a DNS-like distributed cache-based scheme whereby popular sites' userids are cached at a proxy close to a set of users.

A more intriguing way by which site owners can negate the usefulness of registration avoiders and sharing of userids is to increase the value of the userids to each user. Currently, if users are browsing a newspaper site and have not shared any valued information to obtain a userid then they may be willing to contribute it to a public database. There is an interesting privacy *gain* by contributing—if a large number of users use the same userid, then the site cannot gather tailored information that could violate the privacy of any single user who uses that userid. However, if the user's credit card information is available before a userid is given out, the content owner can be more certain that the userid will not be shared with the general public. The New York Times'

TimesSelect mechanism works in this fashion: subscribers are given access to the "premium" section of the newspaper. Sharing that userid allows any one to access subscriber's private features: one can cancel the subscription, change the physical delivery address, or order products that may be available on the site. Naturally the legitimate owners of that userid are not likely to make it available to the general public.

6. RELATED WORK

Prior work such as [8] and [3] has focused on various factors affecting performance in the download of Web pages, although we are not aware of any work that has examined the performance impact of extraneous content.

We are not aware of work preceding BugMeNot that has functionality related to circumventing Web registration. There have been numerous attempts to filter URLs being fetched and enable privacy via anonymous downloading. Prior work has been at the granularity of the URL as opposed to specific sections of a page. Filtering techniques examined the URL strings and in many cases the contents of the URLs. Specific filters to remove advertisements have been available; for example WebWasher Classic [15] claims to eliminate ad banners, pop-up windows, animations, cookies, and suppresses invocation of scripts. Personal proxies (e.g., Ad Buster) and other individual URL-based blockers (e.g., SynergeticSoft) abound. A list of servers that could be avoided has been available for several years. The hosts file project (http://remember.mine.nu) has a list of 40,000 hosts to be blocked and the list is regularly updated. The idea is that when the supposed ad servers are looked up they are redirected to an internal address avoiding any contact with the ad server. A variant is to use Proxy auto-configuration browser setting to trigger the browser to contact a non-existent proxy when ads are located. Some blockers are in the form of Cascading Style Sheets (e.g., Floppymoose (http://www.floppymoose.com), although this tool downloads ads, but does not display them.

7. SUMMARY AND FUTURE WORK

Our study has shown that extraneous content exists on the majority of a set of popular pages and that a 25-30% reduction in downloaded objects and bytes with corresponding latency reduction can be attained by blocking it. Numerous filters have been written and shared for blocking ads with a popular Firefox browser extension. We found that the top ten ad-delivering companies delivered 40% of all ad URLs in our study. We also found that both the server name and the rest of the URL are equally important to be matched as ads. We found that the majority of ads could be blocked simply by disabling JavaScript, but that is impractical given the dependency on it for accessing many sites. In addition, we found that a majority of these popular sites require some form of registration and for those that do require registration, most have shared userids available in the BugMeNot public database. The concerns of content owners, advertisement servers, and a subset of users are in clear conflict. The significant latency reduction, avoiding hassles in registering for rarely used sites, and privacy preservation are a strong incentive for users to resort to these public domain tools.

The risk of alienating legitimate readers and the fact that a vast majority of visitors to a site are not using public database userids are the primary reasons why content owners have thus far not bothered to take active measures. The registration avoiders are further likely to give less than honest information if they were to fill out the forms. When popular sites that depend on being able to identify individual users notice a significant increase in popularity of an avoidance technique, they may resort to blocking and blacklisting measures. Coupled with increasing prevalence of ad blocking they may not have a choice in the cat and mouse game.

We plan to continue to monitor this tussle between content owners and users with possible changes required in our methodology. Enhancements to our methodology, such as monitoring longer sessions at a site instead of single page visits is also of interest. The impact of content personalization and the use of cookies by advertisement servers are other topics we plan to explore.

8. ACKNOWLEDGMENTS

We thank the creator of BugMeNot (Anonymous) for data and additional information related to BugMeNot. We thank Harsha Madhyastha's help with the PlanetLab data collection and his comments on the paper. Our thanks to Aaron Archer, Howard Karloff, Jeff Mogul, Stephen North, and Wladimir Palant for their comments on the paper.

9. REFERENCES

[1] Adblock. http://adblock.mozdev.org/.

[2] Alexa: Most popular web sites. http://www.alexa.com/.

[3] L. Bent and G. Voelker. Whole page performance. In *Proceedings of the 7th International Workshop on Web Content Caching and Distribution*, Boulder, Colorado, August 2002.

[4] bugmenot.com, tell everyone you know. http://www.bugmenot.com/.

[5] E. Cohen and B. Krishnamurthy. A short walk in the Blogistan. *Computer Networks*, Spring 2005.

[6] Official home of Filterset.G. http://www.pierceive.com/.

[7] O. Kharif. Why web publishers fear a little sharing. BusinessWeek Online. October 19 2004 http://www.businessweek.com/bwdaily/dnflash/oct2004/nf20041019_9800_db016.htm.

[8] B. Krishnamurthy and C. E. Wills. Analyzing Factors that influence end-to-end Web performance. In *Proceedings of the WWW*, pages 17–32, May 2000.

[9] B. Krishnamurthy and C. E. Wills. Improving Web Performance by Client Characterization Driven Server Adaptation. In *Proceedings of the WWW*, May 2002.

[10] R. Kumar, J. Novak, and P. Raghavan. On the Bursty Evolution of Blogspace. In *Proceedings of the WWW*, 2003.

[11] R. Metz. We don't need no stinkin' login. WIRED Magazine, July 2004 http://www.wired.com/news/print/0,1294,64270,00.html.

[12] Remote control of unix mozilla. http://www.mozilla.org/unix/remote.html.

[13] Planetlab. http://www.planet-lab.org/.

[14] 100 of the most popular political websites on the net. http://www.rightwingnews.com/special/rank.php.

[15] Webwasher classic. http://www.cyberguard.com/products/webwasher.