Greening the Internet: Measuring Web Power Consumption

Aruna Prem Bianzino, Anand Kishore Raju, and Dario Rossi, Telecom ParisTech

This evaluation of end-user PCs browsing the Web—and loading Flash plug-ins—measures power consumption by considering the hardware platform, operating system, browser, and website. It also reveals the unnecessary power expenditures of tabbed browsing.

With the mass-market adoption of networking technology, the Web has often been used as a synonym for the Internet. In recent years, renewed user interest in the Web comes from cloud services such as GoogleDocs, multimedia-rich portals such as YouTube and Megavideo, and social networking sites such as Facebook and Twitter.

Although we still refer to this rich set of applications simply as the Web, clearly much has changed in terms of the underlying technology. On the client side, Adobe Flash is one example of this evolution—and perhaps the most important one, given its penetration rate of almost 100 percent in some markets. Some have criticized Flash for being CPU hungry, resulting in comparative tests against alternatives such as Silverlight and HTML5. Other studies have measured the average power consumption of websites (focusing on dynamic content or battery lifetime), thereby providing anecdotal evidence of the existing correlation between CPU load and power usage. Along similar lines, we evaluated the power consumption of the Web from the end-user viewpoint, considering the variability of the websites, browsers, operating system, and hardware equipment.

Methodology

The power drain $P$ of any end-user device can be divided into a fixed component $P_0$—representing power consumption in an idle state—and a variable component proportional to the sustained workload $\alpha$—that is, $P = P_0 + f(\alpha)$. An experimental methodology to assess the power drain must consider several factors, including the hardware, operating system, browser settings, and benchmark websites. The fixed component can be affected by both the hardware (such as CPU frequency, voltage scaling, and video-card graphic
processing unit) and the operating system (such as a tickless kernel). Web software and content affect the variable component; for example, webpage scripts generate different workloads depending on the complexity of the tasks to be performed. Web content and software can also interact with other factors; for example, software can be optimized for a specific operating system.

We ran experiments on three PC platforms: a Dell desktop PC and Dell and Apple Mac laptops. As Table 1 shows, the system configurations included three operating systems (Windows Vista, OS X, and Ubuntu Linux), and four browsers (the most recent stable versions of Internet Explorer, Firefox, Chrome, and Safari). We used two browser settings (Flash enabled and disabled) and tested 14 websites (see Table 2). We repeated each experiment twice (resulting in more than 500 experiments), browsing to a specific webpage and measuring the power drain on our end system during a fixed timeframe. Data is available for the energy consumed by data centers and Internet data transfers, but we didn’t account for it in our experiments (interested readers should instead refer to the cited articles).

In measuring power consumption, we tried to balance platform variability (hardware, operating system, and software settings) with our benchmark websites. We considered 18 platform configurations, avoiding configurations such as Safari over Windows on the desktop PC, or Internet Explorer over Linux on the laptop. Although we didn’t explore the full cross product of settings, the ones we chose should well represent both households and enterprises.

Other researchers have performed similar studies. For example, Robert Hansen tested approximately 100 websites but only on a single hardware platform and operating system (for a total of four platform settings). Jarred Walton,

### Table 1. Platform heterogeneity—the hardware, operating system, and software specifications and configurations used in the experiments.

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Product</th>
<th>Configuration</th>
<th>Operating System</th>
<th>Browsers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debian</td>
<td>Desktop</td>
<td>Dell OptiPlex GX240</td>
<td>512 Mbytes RAM, Intel Pentium 4, 1.50 Ghz</td>
<td>Ubuntu v. 10.4 LTS Chrome v.5.D.375.B6 Firefox v.3.6.3</td>
</tr>
<tr>
<td>Lapdeb</td>
<td>Laptop</td>
<td>Apple MacBook Pro</td>
<td>2048 Mbytes RAM, Intel Core 2 Duo, 2.50 Ghz</td>
<td>Ubuntu v. 10.4 LTS Chrome v.5.0.307.9 Firefox v.3.6.3</td>
</tr>
<tr>
<td>Mac</td>
<td>Laptop</td>
<td>Apple MacBook Pro</td>
<td>2048 Mbytes RAM, Intel Core 2 Duo, 2.50 Ghz</td>
<td>OS X v10.6.2 Chrome v.5.0.307.9 Safari v.4.0.4</td>
</tr>
<tr>
<td>MSV</td>
<td>Laptop</td>
<td>Dell Vostro</td>
<td>2048 Mbytes RAM, Intel Core 2 Duo, 1.60 Ghz</td>
<td>Windows Vista, SP2 Chrome v.5.D.375.B6 Firefox v.3.6.3 Explorer v8.0.6001.18928</td>
</tr>
</tbody>
</table>

### Table 2. Web heterogeneity—the websites tested.

<table>
<thead>
<tr>
<th>Website</th>
<th>Alexa ranking</th>
<th>Content type</th>
<th>No. of Flash plug-ins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gmail</td>
<td>1</td>
<td>Email client</td>
<td>0</td>
</tr>
<tr>
<td>Wikipedia</td>
<td>6</td>
<td>Information portal</td>
<td>0</td>
</tr>
<tr>
<td>YouTube</td>
<td>3</td>
<td>Video streaming</td>
<td>1</td>
</tr>
<tr>
<td>Amazon</td>
<td>19</td>
<td>eCommerce</td>
<td>1</td>
</tr>
<tr>
<td>Collegehumor</td>
<td>1,251</td>
<td>Entertainment blog</td>
<td>1</td>
</tr>
<tr>
<td>Fnac</td>
<td>1,699</td>
<td>eCommerce</td>
<td>1</td>
</tr>
<tr>
<td>Spotify</td>
<td>2,834</td>
<td>Music streaming</td>
<td>1</td>
</tr>
<tr>
<td>BBC</td>
<td>40</td>
<td>News portal</td>
<td>2</td>
</tr>
<tr>
<td>Yahoo</td>
<td>4</td>
<td>Online portal</td>
<td>3</td>
</tr>
<tr>
<td>CNN</td>
<td>58</td>
<td>News portal</td>
<td>3</td>
</tr>
<tr>
<td>Mashable</td>
<td>292</td>
<td>Technical blog</td>
<td>4</td>
</tr>
<tr>
<td>Slashfilm</td>
<td>4,403</td>
<td>Review blog</td>
<td>5</td>
</tr>
<tr>
<td>LeMonde</td>
<td>769</td>
<td>News portal</td>
<td>6</td>
</tr>
<tr>
<td>Gazzetta</td>
<td>585</td>
<td>Sports portal</td>
<td>7</td>
</tr>
</tbody>
</table>
on the other hand, considered more browser appliances (seven browsers on three hardware/operating system platforms for 21 different platform settings) but for only three websites (technical blogs). So although Hansen and Walton have performed detailed analyses concerning a single aspect, they miss the overall picture—that is, the relative importance of each factor.

Platform Heterogeneity
Figure 1 reports preliminary results to show the differences that arise due to hardware, operating system, and software configuration. To estimate an upper bound of the potential differences, we performed a stress test by simultaneously playing five high-quality videos from YouTube. We repeated each test twice, cleaning the browser cache prior to each execution to account for the content download.

Each test lasted 120 seconds, and every 10 seconds we measured the power consumption (with a wattmeter) and the CPU load. Aside from the browser being tested, only a task manager (Windows) and shell (OS X and Linux) processes were running (to measure the CPU load).

Figure 1 reports the average CPU load during the stress test and the corresponding average power consumption. For reference, we also include the power usage of the idle system ($P_0$), without any browser. For each hardware/operating system configuration, the most power-efficient browser appears first. We also report the percentage increase with respect to the baseline browser.

Although the CPU load varied between 70 and 100 percent for all platforms, the power-consumption range (25 to 85 W) was wider because of the desktop PC’s inclusion in the experiments (note that the wattmeter didn’t measure the external LCD screen’s plug). The CPU load can saturate to 100 percent; at the same time, different operating systems for a given hardware (for example, Linux vs. OS X for the MacBook) or different software for a given operating system (Chrome or Firefox for Windows) can bring the CPU load below 100 percent. The power consumption can rise well above the baseline, so there’s clearly room for substantial power savings.

Website Heterogeneity
It’s difficult to select a set of webpages truly representative of the average use. Hansen considered the 100 most popular websites according to the Alexa ranking and then selected the 10 most CPU hungry sites. You wouldn’t want to use just the Alexa rankings because, for example, although Google is the top-ranked site, its homepage is lightweight by design, occurs more than once in the ranking (due to country-local replicas), and likely represents a first quick hop toward other, more content-rich and CPU-intensive webpages.

We thus carefully selected our websites. We looked for ones pertaining to different services (such as blog, news, video, or music) in an attempt to mimic different user activities. Also, some of the websites aren’t popular worldwide, but they’re very popular in the respective countries (for example, Gazzetta is very popular in Italy, as LeMonde and Fnac are in France). Finally, we carefully selected pages containing differing numbers of Flash plug-ins (from zero for Gmail and Wikipedia to seven for Gazzetta).

For each website, we accessed each page with the different configurations outlined in Table 1.
and recorded the CPU and power metrics for 120 seconds. In all our experiments, we either disabled Flash or used Flash Player 10.0 (the most recent version available at the time). (Note that results might differ—especially in terms of CPU load—with Flash Player 10.1, which is designed to exploit the hardware acceleration capabilities offered by the GPU by transferring the load from the CPU to the GPU. However, we expect results to remain valid in terms of the overall CPU and GPU load.)

Figure 2 presents the results as a scatter plot depicting each hardware, operating system, and software configuration as a circle. Each circle’s center is located at the average CPU load and power consumption, and each radius equals the standard deviation of the CPU load. Due to the importance of video (which is currently rising faster than any other service type and is forecasted to account for over 90 percent of Internet traffic in the next few years), Figure 2a refers only to YouTube, while Figure 2b reports the average performance over all of the websites tested.

High-quality video streaming was the most CPU-hungry Web application, reaching up to 70 percent CPU utilization, while the average CPU load of other services didn’t exceed 25 percent. Also, the CPU rise depended on the specific configuration, as shown by the dispersion of the red circles with respect to the clustered blue ones. Indeed, the absolute power-consumption value depended primarily on the hardware platform, but differences can also arise depending on the specific operating system and software configuration.

**Assessment**

Here, we weighed the relative impact of the platform and websites on the overall end-user power consumption. Then we conservatively quantify the potential absolute power saving that one simple technique could achieve.

**Relative Weight**

To gauge the relative importance of the considered parameters, we measured the extent of load $L$ (or power $P$) variability by evaluating, first, the ratio between the maximum and minimum $L$ (or $P$) values gathered by varying a single parameter at a time and, then, the average over all other parameters.

For example, to measure the website impact on power variability $I_P$, we averaged the maximum to minimum power-ratio over all possible platform settings, gathered over all website $w$ experiments, such that $I_P = E_w[\frac{\max_w P(s)}{\min_w P(s)}]$, where $E_w[·]$ denotes the average over all possible (hardware, operating system, and software) settings $s$.

Because our testbed doesn’t exhaustively explore all platform combinations, the relative importance for some parameters reflects only the representative subset. For the sake of clarity, to
measure the hardware impact, we considered the
desktop PC and Mac laptop, with the Chrome
and Firefox browsers under Debian, averaged
over all websites. The desktop PC consumes
more than the laptop, so it stands as the numera-
tor in

\[
I_r = E_w \left[ \frac{1}{2} P(\text{debian - chrome}) + \frac{1}{2} P(\text{debian - firefox}) \right]
\]

where \( E_w \) denotes the average over all websites.

Table 3 reports ratios for both CPU load and
power consumption, revealing that operating sys-
tem impact is less prominent than other factors.
As expected, hardware is the most important fac-
tor and could also induce a larger variability of
CPU utilization. Interestingly, different software
browsers affect CPU usage, though not enough
to significantly change the power consumption.
(With minor exceptions, Walton’s results point to
similar conclusions.5) The power ratio is lowest
among all factors, indicating that users can stay
with their preferred browser without harming
the planet too much.

Finally, the specific webpage can have a dra-
matic impact on the CPU load, which in turn
translates into a large variability in the power
consumption. In this case, blame could fall upon
both the developers of the Flash scripts (for im-
plementing unoptimized scripts where an ani-
mated image would probably suffice) and of the
core the Flash interpreter (which has far from
optimal performance).

**Absolute Power Saving**

Now let’s quantify the absolute power waste (or
potential gain). A single webpage consumes on
average approximately 4.7 W, which can grow
up to nearly 16 W in the not-so-uncommon
case of video streaming. (Estimates are con-
sistent, although higher, with respect to the
3.3 W to 11.3 W range Hansen gives for a single
platform.4)

Users, while browsing, generally leave numer-
ous tabs open and then close a batch of tabs from
time to time. Unless the number of tabs causes
a noticeable performance slowdown (compelling
users to close some older tabs), users have no real
incentive to keep only one or two tabs open at the
same time. Yet users are likely consulting only a
single tab, so the other open tabs are unnecessar-
ily using the CPU and wasting power.

It’s like our grandmothers always told us,
“Don’t leave the light on when you’re in another
room.” In the world of Web browsers, this would
sound like, “Run only the scripts on the currently
active tab, of the focused window, on the focused
desktop.” Within the browser, this means freez-
ing the execution of all scripts that aren’t on the
focused tab (implemented as sleep/resume signals
sent to the script interpreter on each tab focus
change). We could save even more power if the
browsers would stop executing scripts any time
tabs are hidden from the end user (such as when
the tab is focused but the corresponding window
is iconized or in another nonvisible desktop).

To evaluate this strategy’s lower power-saving
bound, we considered users who are keeping
only two tabs open—one that they’re actively
consulting and another that’s just open. Given
the estimated average consumption per web-
page, this would save on the order of a 10 W
light bulb’s worth of power. Evaluating the up-
per bound requires estimating the number of
tabs that could be left open without causing a
noticeable slowdown. Although the exact CPU
threshold for a noticeable slowdown is subjec-
tive, bounding CPU utilization to 75 to 90 per-
cent, our measurement suggests that seven to 10
tabs could be opened in parallel without a no-
ticeable performance drop for even the slowest
platform. Considering a household with three
computers, the waste could turn into a 100 W
power drain. As countries around the world are
in the process of phasing out 100 W light bulbs
in favor of the low-energy alternative, it seems
reasonable to consider a similar revolution for
Web browsing.

**Web browsing has evolved greatly in
terms of content richness, services of-
fered, and available user interfaces.
Considering client-side dynamically generated**
content, we think another evolution is around the corner—one that will reduce unnecessary power consumption.

A potentially large source of power waste arises due to tabbed browsing, causing several scripts to run in parallel. However, users typically interact with only one tab at any given time, so scripts running on unfocused tabs unnecessarily waste resources. Clearly, our results are far from exhaustive—this would require considering many more platforms configurations and websites. Nevertheless, we hope this article offers a valuable starting point for greener Web browsing.

Acknowledgments
The authors thank their grandmas, whose useful precepts go well beyond the scope of this article.

References

Aruna Prem Bianzino is a PhD student in the Computer Science and Networking Department of Telecom ParisTech. His research interests include the interworking of energy awareness in wired networks, interdomain routing, and quality of service. Bianzino received his MSc from Politecnico di Torino for work developed as an internship project at the Centre Tecnològic de Telecomunicaciones de Catalunya. Contact him at bianzino@enst.fr.

Anand Kishore Raju works as a research engineer in the Computer Science and Networking Department of Telecom ParisTech. His research interests include distributed collaborative computing and green networking. Kishore received his BSc from the National Institute of Technology (NIT) Durgapur, India. Contact him at raju@enst.fr.

Dario Rossi is an associate professor in the Computer Science and Networking Department of Telecom ParisTech. His research interests include green networking, peer-to-peer networks, Internet traffic measurement, and traffic engineering. Rossi received his PhD in electrical engineering from Politecnico di Torino. Contact him at dario.rossi@enst.fr.
This article was featured in computingnow.

For access to more content from the IEEE Computer Society, see computingnow.computer.org.

IEEE computer society

Top articles, podcasts, and more.