radware

STATE OF THE UNION

Ecommerce Page Speed & Web Performance

Winter 2013-14

SHARE THIS WHITEPAPER

Table of Contents

Executive Summary

2013 saw an unprecedented number of headline-grabbing website outages. These incidents made the news because downtime hurts: Amazon.com's 40-minute outage in August 2013 reportedly cost the company up to $5 million in lost sales[1], while Healthcare.gov's availability issues resulted in significant ongoing brand damage, not just to the new healthcare initiative, but to the current Democrat administration.[2]

Site outages are just one face of performance. Just as damaging, but much more insidious, is the impact of site slowdowns on business metrics ranging from online revenues to brand perception.

According to a survey[3] of 300 businesses, respondents estimated average revenue losses of $21,000 per hour of downtime, and $4,100 for every hour of performance slowdown. In the same survey, these companies reported that website slowdowns occur 10 times more frequently than website outages. Based on these numbers, **website slowdowns can have double the negative impact on an organization's revenues as outages.**

The Goal of This Report

Since 2010, we have been measuring and tracking the performance and page composition of leading ecommerce sites. **The purpose of this research is to gain ongoing visibility into the real-world performance of leading ecommerce sites – to learn how these sites perform for visitors using the internet under normal browsing conditions – and to provide strategies and best practices that will enable site owners to serve pages faster.**

This report answers the following questions:

- Given the assumption that page speed is an increasingly urgent issue for online retailers, has this urgency translated into faster pages over time?
- How do pages actually render in real-world scenarios?
- How quickly have retailers moved to adopt core performance best practices, such as using content delivery networks (CDNs)?

Our findings are detailed in this report, and are summarized here:

Key Findings

1. The median page has slowed down by 21% in just one year.
The median top 500 ecommerce home page takes 9.3 seconds to load. A year ago, the median page took 7.7 seconds to load. The majority of online shoppers will abandon a page after waiting 3 seconds for it to load.[4]

2. The top 100 sites are slower than the top 500.
Among the top 100 ecommerce sites, the median load time is 10 seconds – up from 8.2 seconds at this time last year. The maximum threshold that a typical internet user is willing to wait for a page to load is 10 seconds[5], meaning that half of the top 100 retail sites do not meet this threshold.

3. Pages are taking longer to become interactive.
"Time to interact" (TTI) refers to how long it takes for a page's primary content to load and become usable. In 2013, the median TTI was 4.9 seconds. Now it's 5 seconds. Some may not consider this a significant increase, but it will be interesting to see how this trend develops in the future.

4. Pages are now bigger and heavier.
The median ecommerce page contains 99 resources (things like image and CSS files). A year ago, the median page contained 93 resources. The median page is 1436 KB in size, a 31% increase over the median page weight of 1094 KB just one year ago. This growth is partially responsible for the increase in load time.

5. The adoption of some core performance best practices has plateaued.
In spring 2013, 74% of the top 100 ecommerce sites used a content delivery network (CDN): this number has grown to 80%. Keep-alives has plateaued at 93% implementation rate. Image compression is still not widely adopted: implementation rate continues to stand at 9%. While the adoption rate of long-standing best practices has not increased significantly, we did note that the use of progressive JPEGs, a practice that had fallen out of favor but is now on the upswing, has increased from 6% of the top 100 sites to 10%.

Except where otherwise noted, the results discussed in this report are for pages tested in Chrome 31. At the time of conducting this research, Chrome was the most widely used browser in the United States, with a market share of 34%.[6]

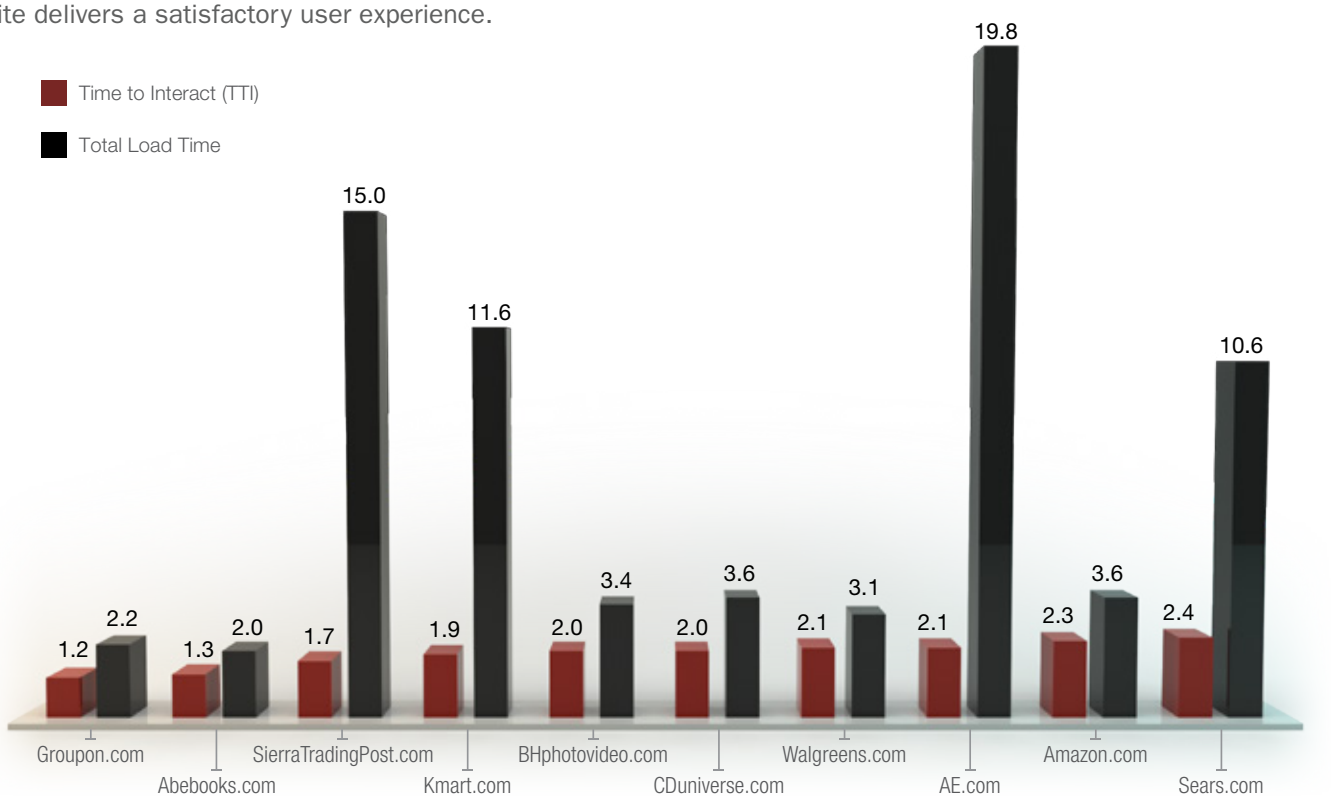
Who Was Fastest?

All page load times are indicated in seconds.

The 10 Fastest Sites (Time to Interact)

Among the top 100 sites, these were the fastest in terms of their ability to display meaningful, interactive content (e.g., feature banners with functional call-to-action buttons). This metric is known as "time to interact" (TTI).

We have provided the TTI alongside the full load time in order to give perspective into the distinction between the two, and to illustrate that **load time is not always the most meaningful measure of a site's performance**. For example, while AE.com has a load time of 19.8 seconds, it has a TTI of 2.1 seconds; the TTI indicates that this site delivers a satisfactory user experience.

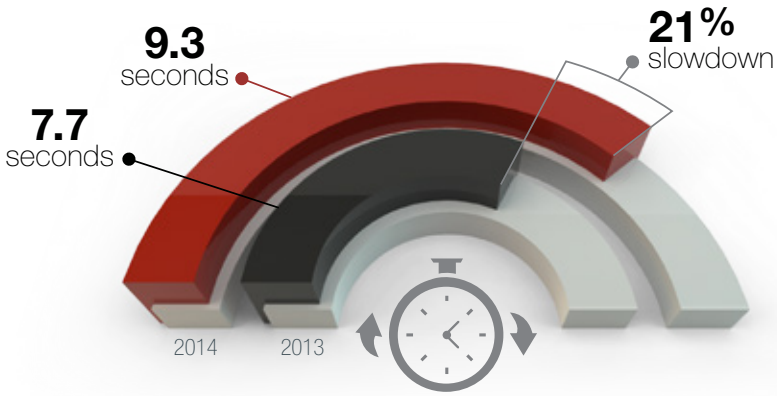


The 10 Fastest Sites (Load Time)

Among the top 100 sites, these were the 10 fastest in terms of the time required for all page elements to load. **It is worth noting** that at the beginning of 2013 there were two sites that loaded in fewer than 2 seconds. By the end of 2013, there were none.

Finding #1: The Median Page Has Slowed Down by 21% in Just One Year

The median top 500 ecommerce home page takes 9.3 seconds to load. A year ago, the median page took 7.7 seconds to load. The majority of online shoppers will abandon a page after waiting 3 seconds for it to load.

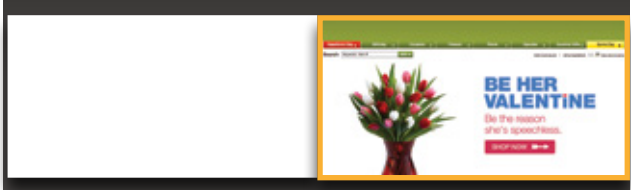


A 21% load time increase in just one year is a significant change. Throughout this report, we'll be addressing some of the issues that could be behind this growth. We'll also offer twelve performance optimization tips that address these issues.

Performance Issue:
Long Page Delays with Blank Browser Windows

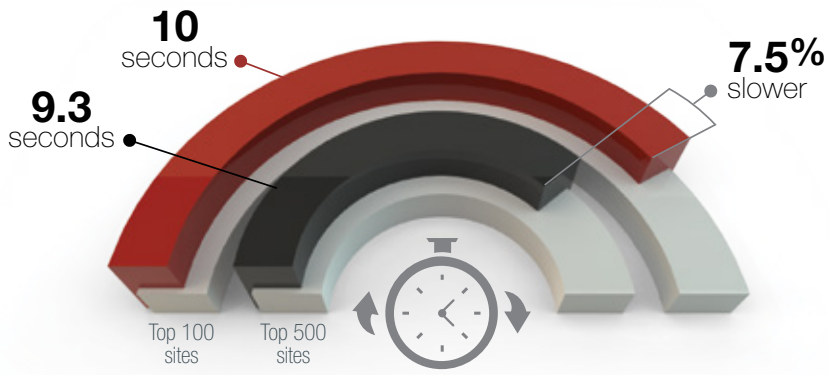
This was the most common design failure. Site owners should be aware of the usability consequence of delaying the rendering of feature content: **a user who endures an 8-second download delay spends only 1% of their total viewing time looking at the featured space on a page**. In contrast, a user who receives instantaneous page rendering spends 20% of their viewing time within the feature area of a page.[7]

7.5s 7.6s

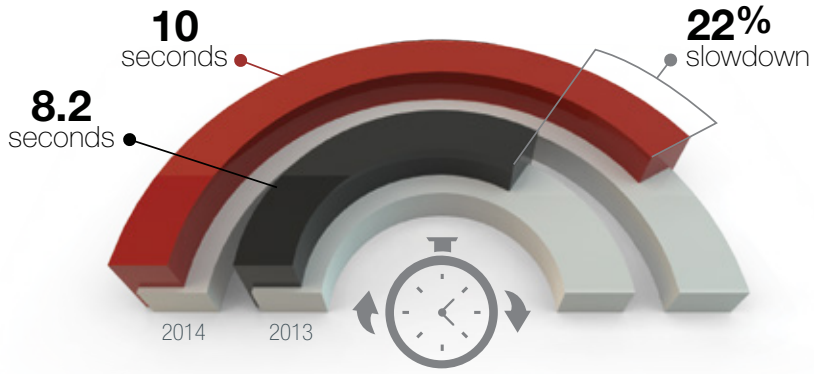


Finding #2: The Top 100 Sites Are Slower Than the Rest of the Pack

We compared load times for the top 100 ecommerce sites to load times for the top 500 sites. Among the top 100 sites, the median load time is 10 seconds – making the median top 100 site 7.5% slower than the median top 500 site (at 9.3s).



As with the top 500 sites, the top 100 sites have also slowed down in the past twelve months. At the beginning of 2013, the median page took 8.2 seconds to load – meaning that the median page is 22% slower now than it was one year ago.

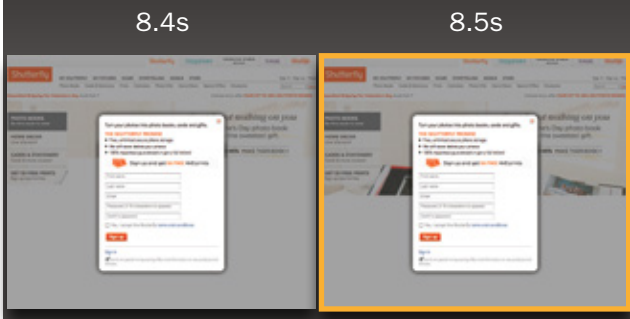


It is important to note that the maximum threshold that a typical internet user is willing to wait for a page to load is 10 seconds[8], meaning that half of the top 100 retail sites do not meet this threshold.

Performance Issue:
Pop-ups That Obscure the Page Before It Renders

This was a recurring usability issue on many sites: within moments of arriving at the home page, users are served with a pop-up. This presents both a usability issue and a performance issue.

There are some use cases that support on-arrival pop-ups, such as requiring a user to identify their location in order to serve accurate item and shipping costs; however, in many cases the pop-ups noted in our study were for newsletters, surveys, and other opt-in marketing campaigns. Not only do these pop-ups act as a usability irritant, they slow down or block the rendering of the main page content.



Why Is This Finding Significant?

There is a general assumption that leading web sites must have fewer performance-related issues than other sites, due to their larger development and operations budgets. In fact, often the opposite is true. As an example, leading sites tend to take a more aggressive approach to incorporating third-party marketing tools, such as trackers and social analytics, and these scripts can have a significant impact on performance. These sites are also more likely to incorporate large, high-resolution images in their design.

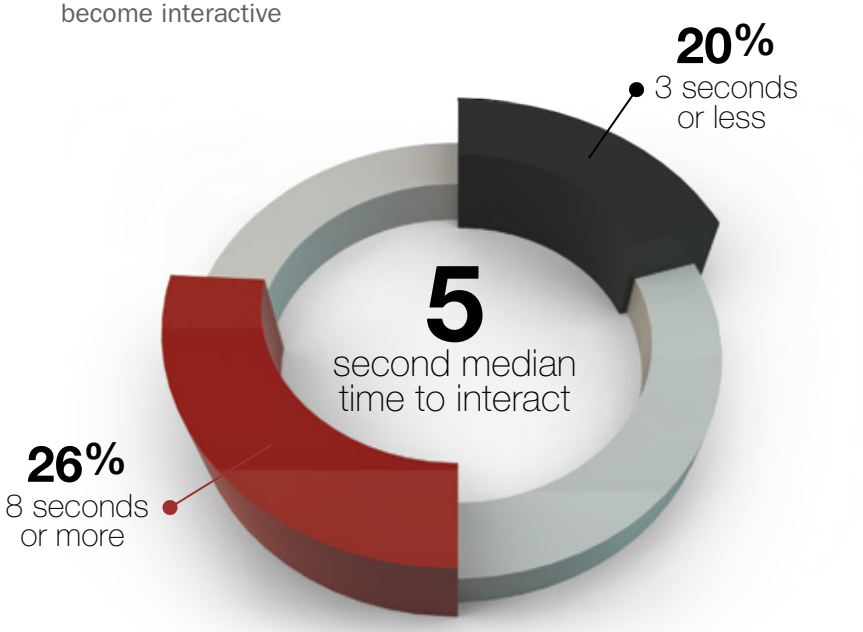
Finding #3: Median Time to Interact Is 5 Seconds

Time to interact (TTI) is the point at which a page displays its primary interactive content (e.g., feature banners with functional call-to-action buttons). TTI is currently the most reliable indicator of a page's ability to fulfill its two primary goals:

1. **Deliver a satisfactory user experience** by delivering content that the user is most likely to care about.
2. **Fulfill the site owner's objective** by allowing the user to engage with the page and perform whatever call to action the site owner has deemed the key action for that page.

To illustrate in measuring the TTI for the top 100 ecommerce sites, we found the following:

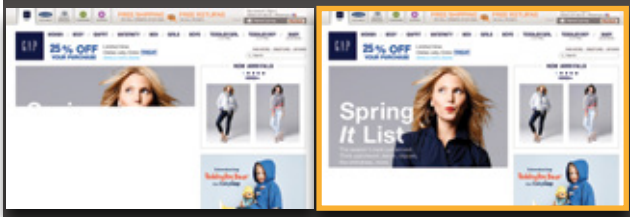
- The median time to interact was 5 seconds
- Only 20% of the top 100 sites had a TTI of 3 seconds or less
- 26% of the top 100 sites took 8 seconds or more to become interactive



Performance Issue:
Call-to-Action Loads Last

Placing the call to action (CTA) at the bottom of feature banners is a widely followed design convention, yet it frequently poses a significant performance penalty. In many of the pages we studied, it was noted that the CTA – arguably the most critical page element – was often the last visible element to render.

8.5s 8.6s



Why Should Pages Be Interactive in 3 Seconds or Less?

Separate studies have found that:

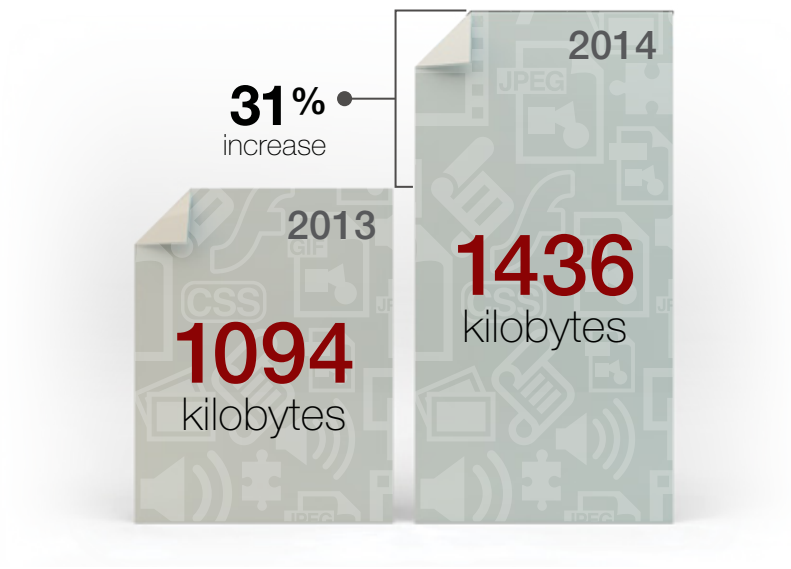
- 57% of consumers will abandon a page that takes longer than 3 seconds to load.[9]
- A site that loads in 3 seconds experiences 22% fewer page views, a 50% higher bounce rate, and a 22% fewer conversions than a site that loads in 1 second, while a site that loads in 5 seconds experiences 35% fewer page views, a 105% higher bounce rate, and 38% fewer conversions.[10]
- Users who have to endure an 8-second download delay spend only 1% of their total viewing time in the feature slideshow space. In contrast, users who receive instantaneous page rendering spend 20% of their viewing time within the feature content space.[11]

Finding #4: Pages Are Now Bigger and Heavier

Looking at the current size and composition of the median ecommerce page, it is easy to note one of the causes of this page slowdown: web pages are growing larger and more complex.

The median ecommerce page contains 99 resources (e.g., images, CSS files, JavaScript). A year ago, the median page contained 93 resources. Each of these resources incurs latency, and this latency adds up to slower load times.

The median page is 1436 KB in size, a huge increase – 31% to be precise – over the median page weight of 1094 KB just one year ago.

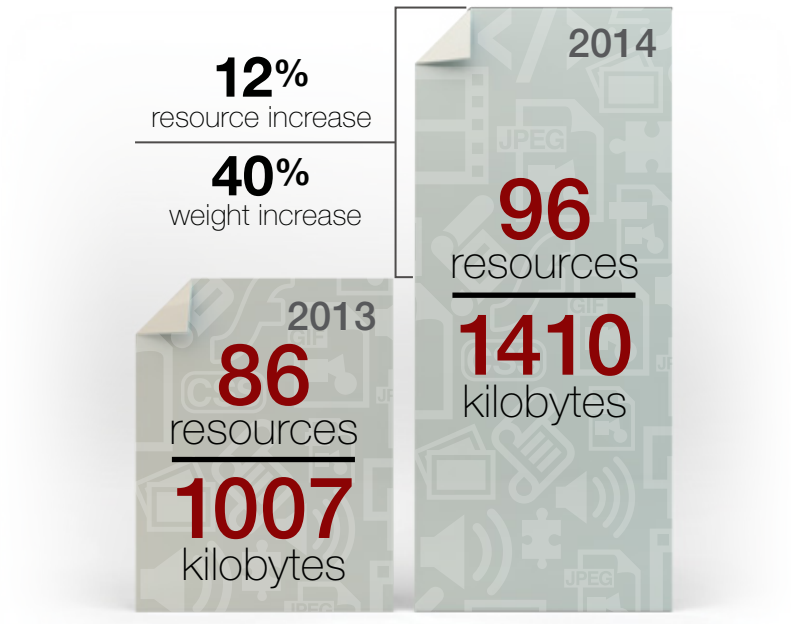


To put it another way, imagine what would happen if you gained 31% of your current body weight and then tried to put on your pants. This is roughly what it's like trying to stuff fatter pages through an Internet pipe that has stayed the same size.

"Page Bloat" Is Even More Pronounced Among the Top 100 Sites

A year ago, the median number of resource requests for a Top 100 site was 86. That number has grown by 12% to 96 resources.

Not only has the number of resources grown, the weight of those individual resources has also increased. In early 2013, the median Top 100 page was 1007 KB. Today, the median page carries a payload of 1410 KB – a surprising 40% gain.



Why Does Page Size Matter?

For most sites, one of the greatest performance drains is the need to complete dozens of network round-trips to retrieve resources such as style sheets, scripts, and images. Each of these resources makes an individual round trip from the user's browser, which requests the file from the host server, which in turn delivers the file to the browser. Each round trip can take between 65 and 145 milliseconds (or longer) for desktop browsers – numbers that add up quickly when a typical page contains almost 100 resources.

Finding #5: The Adoption of Core Performance Best Practices Has Plateaued

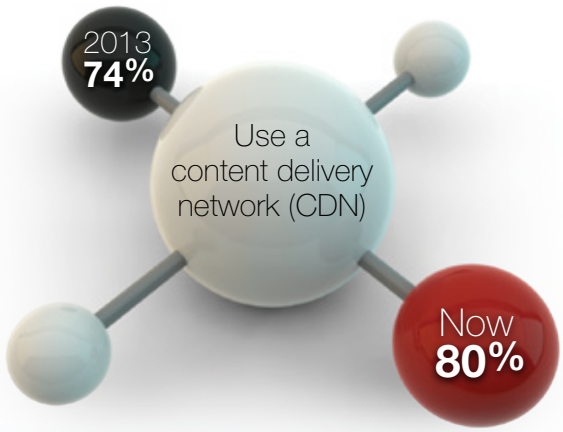
There are a handful of long-standing web performance best practices that represent opportunities to make relatively easy performance gains. Among the top 100 sites, adoption of some of these best practices is nearing the saturation point, whereas others are still neglected. (Letter grades refer to the Google Page Speed scores for each technique. See the legend in this section for details.)

In spring 2013, 74% of the top 100 ecommerce sites used a content delivery network (CDN): this number has grown to 80%. Keep-alives has plateaued at 93% implementation rate. Image compression is still not widely adopted: implementation rate continues to stand at 9%. While the adoption rate of long-standing best practices has not increased significantly, we did note that the use of progressive JPEGs, a practice that had fallen out of favor but is now on the upswing, has increased from 6% of the top 100 sites to 10%.

Use a Content Delivery Network

Using a content delivery network (CDN) to cache page resources closer to end users can shorten server round trip time. This can reduce load time by up to 30%.

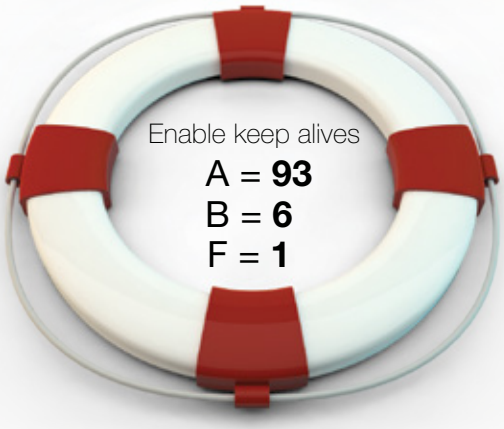
In early 2013, 74% of the top 100 ecommerce sites used a CDN. This number has grown to 80%.



Enable Keep-Alives

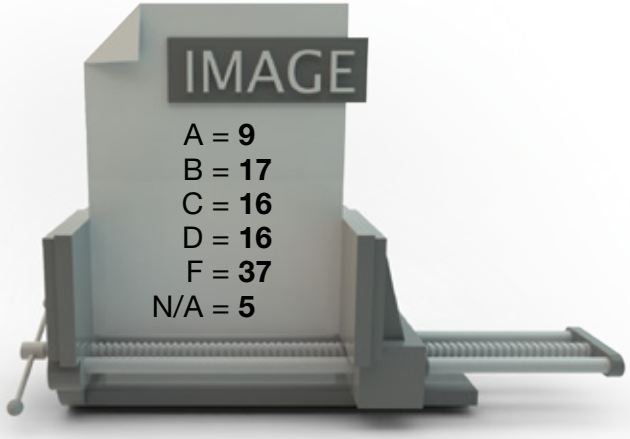
Enabling keep-alives is a performance technique that allows site owners to control how many times the TCP (transmission control protocol) connection takes place. TCP connection is the process by which both the user and the server send and receive acknowledgment that a connection has been made and that data can begin to be transferred. Too many TCP connections will slow down a site. Enabling keep-alives can be as simple as ensuring that your servers and load balancer are configured to do so.

From 2013 to the present, keep-alives has remained consistent with an implementation success rate of 93%.



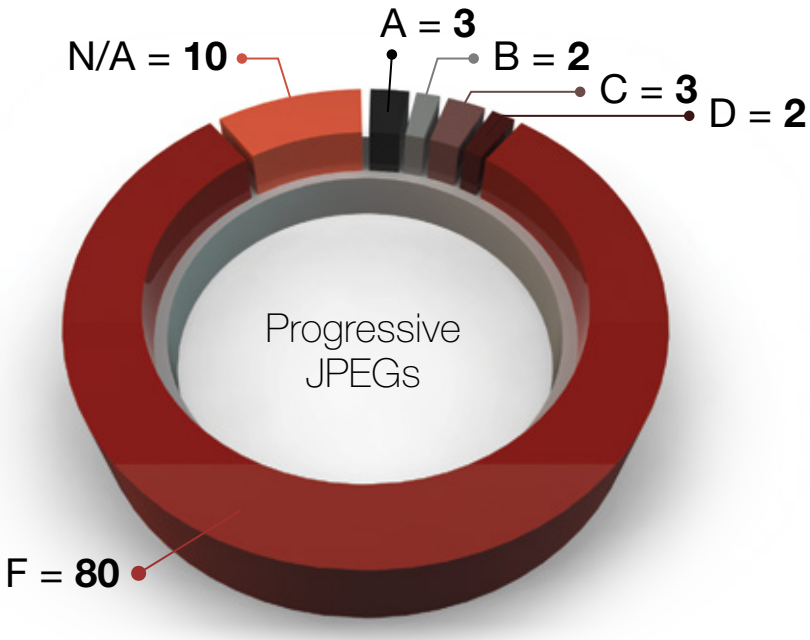
Compress Images

Images comprise the bulk of a typical page's total size, and much of this bulk is unnecessary – caused by poorly optimized or unoptimized images. Image compression is a performance technique that allows site owners to reduce this payload; however, of the top 100 sites we tested, 37% failed to compress images, and only 9% scored an 'A' grade.



Progressive JPEGs

There are two types of JPEGs: baseline and progressive. A baseline JPEG is a full-resolution image that loads in a single top-to-bottom scan. A progressive JPEG is an image that loads in a series of scans, beginning with a low-resolution image and progressing at increasing resolutions until the full resolution is achieved. In many cases, progressive JPEGs improve perceived load time because the user receives visual feedback earlier than with a baseline JPEG.



Only 3% of the sites we tested scored an 'A' on their usage of progressive JPEGs, while 80% scored an 'F'.

Why Do Some Best Practices Thrive While Others Struggle?

To answer this question, first one has to understand the domain that each of these best practices falls into.

The most widely adopted best practices fall outside the web development/content management domain:

- Use a CDN – For all but the very largest websites, content delivery is outsourced to a CDN provider.
- Enable keep-alives – This is a server/load balancer configuration task performed by IT.

The less widely adopted best practices discussed in this report fall within the web development/content management domain:

- Image compression
- Progressive JPEGs

The poor adoption rate of these best practices among developers and content editors can be attributed to a number of factors:

- lack of understanding about the impact of unnecessarily large image files on performance,
- lack of resources dedicated to ongoing performance optimization,
- lack of control over page resources (e.g. due to content management systems), and/or
- lack of investment in front-end optimization tools that automate the implementation of these practices.

Google™

Google Page Speed Score Legend

Google Page Speed assigns sites a score out of 100 for each best practice it measures.

A: 90-100 **B: 80-89** **C: 70-79**
D: 60-69 **F: 0-59**

12 Things You Can Do to Fix Your Performance Pains

There are a number of best practices site owners can implement in order to improve both the real and perceived user experience for online shoppers.

1. Consolidate JavaScript and CSS

Consolidating JavaScript code and CSS styles into common files that can be shared across multiple pages should be a common practice. This technique simplifies code maintenance and improves the efficiency of client-side caching. In JavaScript files, be sure that the same script isn't downloaded multiple times for one page. Redundant script downloads are especially likely when large teams or multiple teams collaborate on page development.

2. Sprite Images

Spriting is a CSS technique for consolidating images. Sprites are simply multiple images combined into a rectilinear grid in one large image. The page fetches the large image all at once as a single CSS background image and then uses CSS background positioning to display the individual component images as needed on the page. This reduces multiple requests to only one, significantly improving performance.

3. Compress Text and Images

Compression technologies such as gzip reduce payloads at the slight cost of adding processing steps to compress on the server and decompress in the browser. These operations are highly optimized, however, and tests show that the overall effect is a net improvement in performance. Text-based responses, including HTML, XML, JSON (JavaScript Object Notation), JavaScript, and CSS, can all be reduced in size by as much as 70%.

4. Defer Rendering "Below the Fold" Content

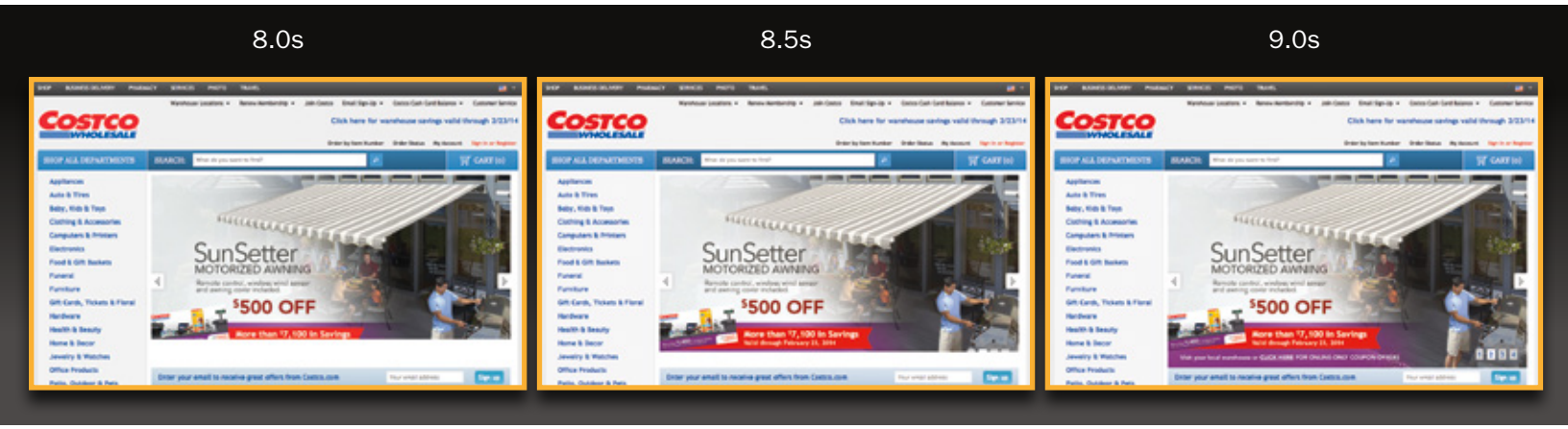
Assure that the user sees the page quicker by delaying the loading and rendering of any content that is below the initially visible area, sometimes called "below the fold." To eliminate the need to reflow content after the remainder of the page is loaded, replace images initially with placeholder <img> tags that specify the correct height and width.

5. Ensure That Feature Images Are Optimized to Load Early and Quickly

As the filmstrip view below shows, the feature image on the Souq.com home page doesn't render until the 10-second mark.

6. Rethink the Design and Location of Call-to-Action Links in Feature Graphics

While the accepted design convention has been to position CTA buttons at the bottom of feature banners, this convention does not always serve the best interests of users or site owners, as shoppers must wait for the image to fully render before taking their next action on the page. In this example, the online coupon code takes almost 9 seconds to render.



7. Reformat Images

Inappropriate image formatting is an extremely common performance culprit. An image that is saved to the wrong format can be several times larger than it would be if saved to the optimal format. Images with unnecessarily high resolution waste bandwidth, processing time, and cache space.

As a general rule of thumb, these are the optimal formats for common image types:

- **Photos** – JPEG, PNG-24
- **Low complexity (few colors)** – GIF, PNG-8
- **Low complexity with transparency** – GIF, PNG-8
- **High complexity with transparency** – PNG-24
- **Line art** – SVG

8. Use Progressive JPEGs

Progressive JPEGs are not a new innovation. They were widely used in the 1990s, but fell out of favor due to performance issues caused by slow connection speeds and crudely rendered JPEGs; watching a progressive image load pixel by pixel was a painful experience. Now that connection speeds have improved and the way in which progressive JPEGs are created has become more sophisticated, this technique is feasible again and is returning as a newly heralded performance best practice. In one study of the top 2,000 retail sites, progressive JPEGs improved median load time by 15%.

(Note: While all popular browsers will render progressive images, Safari, Mobile Safari, Opera and Internet Explorer 8 render them only as baseline JPEGs, meaning there is no performance benefit.)

9. Minify Code

Minification, which is usually applied to scripts and style sheets, eliminates non-essential characters such as spaces, newline characters, and comments. A correctly minified resource is used on the client without any special processing, and file-size reductions average about 20%. Script and style blocks within HTML pages can also be minified. There are many good libraries available to perform minification, often along with services to combine multiple files into one, which additionally reduces requests.

10. Defer Loading and Executing Non-Essential Scripts

Many script libraries aren't needed until after a page has finished rendering. Downloading and parsing these scripts can safely be deferred until after the onload event. For example, scripts that support interactive user behavior, such as drag and drop, can't possibly be called before the user has even seen the page. The same logic applies to script execution. Defer as much as possible until after onload instead of needlessly holding up the initial rendering of the important visible content on the page.

The script to defer could be your own or, often more importantly, scripts from third parties. Poorly optimized scripts for advertisements, social media widgets, or analytics support can block a page from rendering, sometimes adding precious seconds to load times.

11. Use of AJAX for Progressive Enhancement

AJAX (Asynchronous JavaScript and XML) is a technique for using the XHR (XMLHttpRequest) object to fetch data from a Web server without refreshing the page where the code is running. AJAX enables a page to display updated data in a section of a page without reconstructing the entire page. This is often used to respond to user interaction, but it can also enable your application to load a bare-bones version of a page quickly, and then to fill in more detailed content while the user is already viewing the page.

12. Use a Content Delivery Network

Content delivery networks (CDNs) can help performance by bringing content geographically closer to users, thereby shortening round trips between the user's browser and the host server.

Takeaways

1. Performance is still a critical issue for ecommerce sites, and leading sites are not immune.

The ongoing increase in both (TTI) and load time indicates that site owners need to continue their efforts to improve real and perceived load time for their visitors.

2. Page growth shows no signs of slowing down.

The ever-growing use of high-resolution images and third-party scripts means that pages are on a continual upward trajectory. Site owners must look elsewhere to mitigate the impact of design and marketing demands.

3. There are untapped web site acceleration opportunities.

Currently, most site owners rely on their content delivery network and a handful of established core performance treatments to accelerate their pages. Site owners who wish to make further acceleration gains must dig deeper for solutions. There are numerous additional techniques – such as deferral, progressive images, text/image compression, resource consolidation, resource pre-fetching, and browser cache optimization – which can simplify and streamline web pages in realtime, customizing them to meet the unique needs of every browser type, and ultimately accelerate the perception of page speed from the end user's perspective.

4. Site owners need to understand how their pages perform in the real world.

It is not enough to rely on spreadsheets and aggregate performance metrics to understand how a site performs. To truly know how visitors see a site – and to identify usability issues that might otherwise be missed – it is crucial to scrutinize how pages perform frame by frame, and over a real-world connection.

Methodology

The tests in this study were conducted using an online tool called WebPagetest – an open-source project primarily developed and supported by Google – which simulates page load times from a real user's perspective using real browsers. Radware tested the home page of every site in the Alexa Retail 500 nine consecutive times. (The system clears the cache between tests.) The median test result for each home page was recorded and used in our calculations.

The tests were conducted between January 16-26, via the WebPagetest.org server in Dulles, VA, using the latest version of Chrome (31.0) on a DSL connection.

In very few cases, WebPagetest rendered a blank page or an error in which none of the page rendered. These instances were represented as null in the test appendix. Also, in very few cases, WebPagetest.org rendered a page in more than 60 seconds (the default timeout for webpagetest.org). In these cases, 60 seconds was used for the result instead of null.

To identify the time to interact (TTI) for each page, we generated a timed filmstrip view of the median page load for each site in the Alexa Retail 100. TTI is defined as the moment that the featured page content and primary call-to-action button or menu was rendered in the frame.

About Radware

Radware (NASDAQ: RDWR), is a global leader of application delivery and application security solutions for virtual and cloud data centers. Its award-winning solutions portfolio delivers full resilience for business-critical applications, maximum IT efficiency, and complete business agility. Radware's solutions empower more than 10,000 enterprise and carrier customers worldwide to adapt to market challenges quickly, maintain business continuity, and achieve maximum productivity while keeping costs down. For more information, please visit www.radware.com.

Sources

1 VentureBeat, Amazon website goes down for 40 minutes, costing the company $5 million, August 2013
2 Bloomberg, How the iPod President Crashed: Obama's Broken Technology Promise, October 2013
3 PhoCusWright, Consumer Response to Travel Site Performance, June 2010
4 Jakob Nielsen, Website Response Times, June 2010
5 StatCounter Global Browser Stats, December 2013
6 Website Response Times
7 Ibid.
8 Consumer Response to Travel Site Performance
9 Strangeloop Networks, The 90-Minute Optimization Life Cycle, June 2010
10 Website Response Times
11 Patrick Meenan, Progressive JPEGs FTW!, June 2013