Web Performance Today

radware | strangeloop
Now part of Radware

Waterfalls 101: How to understand your website's performance via waterfall chart

9 Jul 2010

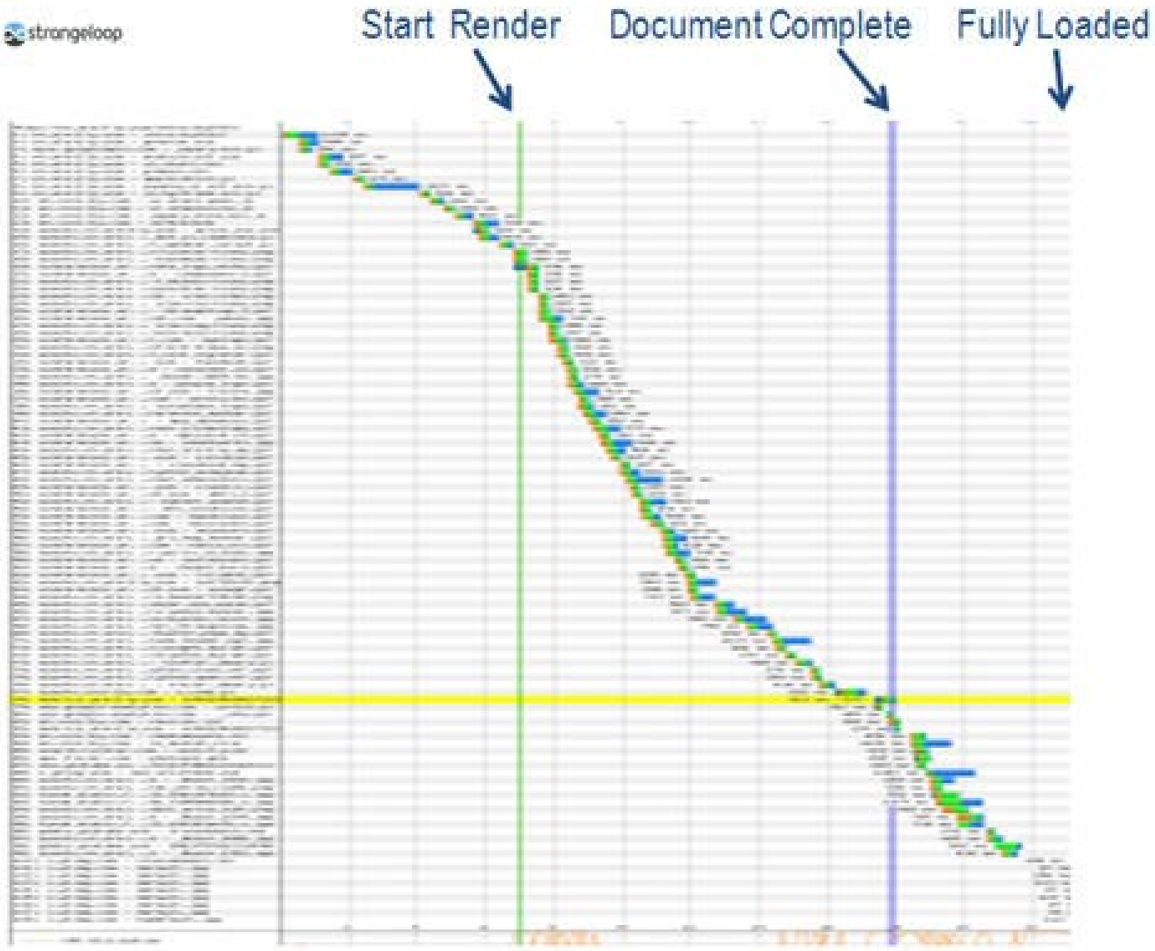
[This post later served as a jumping-off point for a 30-minute webinar: Waterfalls 101: How to read a waterfall chart. http://www.strangeloopnetworks.com/resources/webinars/waterfalls-101-how-to-read-a-waterfall-chart-2/**]**

I keep running into executives who are being presented waterfall charts and don't know how to interpret them. Today I thought it would be a good idea to take a bird's eye view of a typical performance waterfall – pre- and post-acceleration – that you can take away for your own reference or pass along to anyone you think could benefit from having this information.

(If you're technical, live and breath waterfalls, and never plan to speak with an executive, stop reading now. 😐)

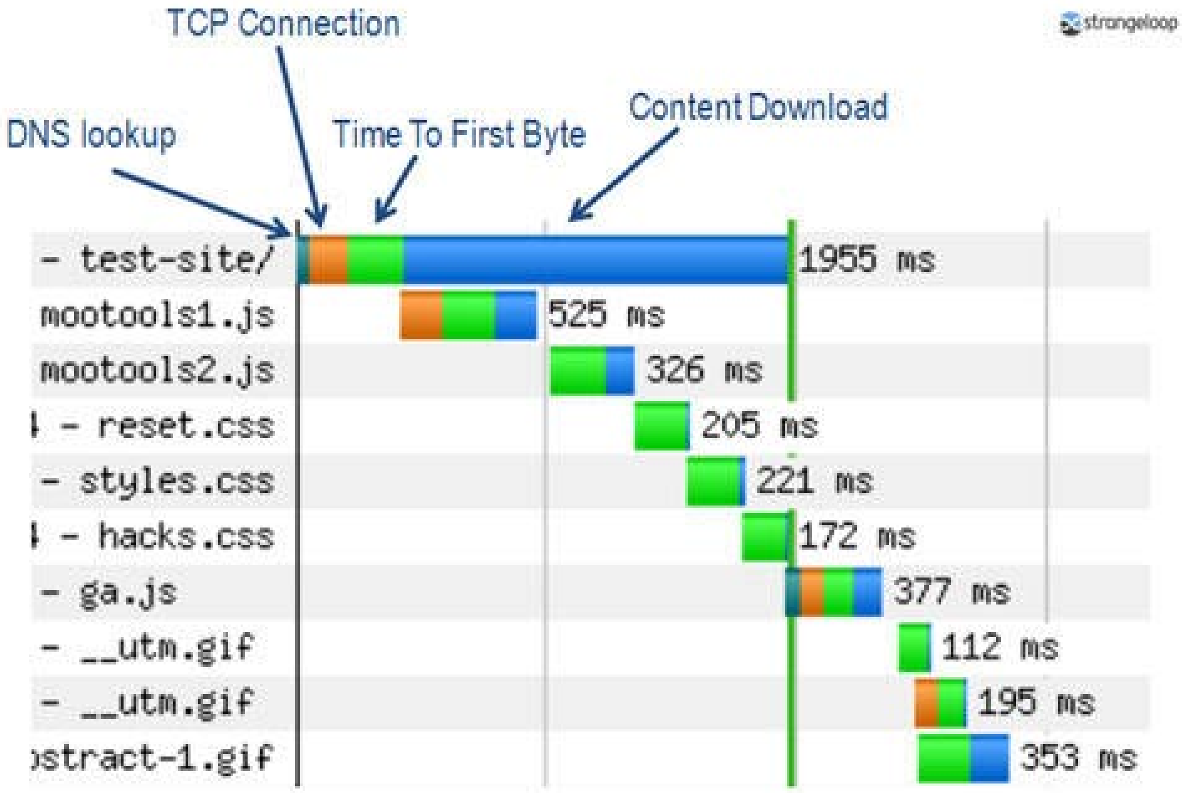
Waterfall charts are diagrams that let you visualize data that is generated cumulatively and sequentially across a process. In the case of performance waterfalls, they let you see **the series of actions that occur between a user and your server in order for that user to view a specific page of your site**.

Here's what a waterfall looks like for a site* that has not been optimized in any way:



To the uninitiated, this is a mess. For now, it's sufficient to understand that each of the rows represents a different object – including text, image, Javascript files, etc. – that the page contains. Depending on whom you ask, **the average web page contains anywhere between 44 and 75 objects** (including text, images, and Javascript). Each of these objects makes its own roundtrip between the server and the browser.

Now let's go in for a closer look:



Each of the colored bars represents a different activity that happens as the object is delivered to the user's browser:

Dark green = DNS lookup

This is when the browser looks up the domain of the object being requested by the browser. Think of this as asking the "phone book" of the internet to find someone's phone number using their first and last name. You can't do much about the dark greens bars and they should not be a problem on most sites.

Orange = TCP connection

Also called the "three-way handshake," this is the process by which both the user and the server send and receive acknowledgment that a connection has been made and data can begin to be transferred. It's sort of like how CB radio users start a conversation:

"Breaker 1-9. This is Bluebeard. Anyone out there read me? Over."
"I read you, Bluebeard. This is Jolly Roger. Over."
"Okay, let's QSO."

(Important note: Everything I know about CB slang I learned from watching *The Dukes of Hazzard*.)

It's not easy to speed up TCP connection, but you can control how many times the TCP connection takes place. Too many connections will slow down site performance.

Bright green = Time to first byte

This is the window of time between asking the server for content and starting to get the first bit back. Continuing our CB radio analogy, think of this as the interval between asking a question to a friend over a radio and waiting to hear the beginning of their response.

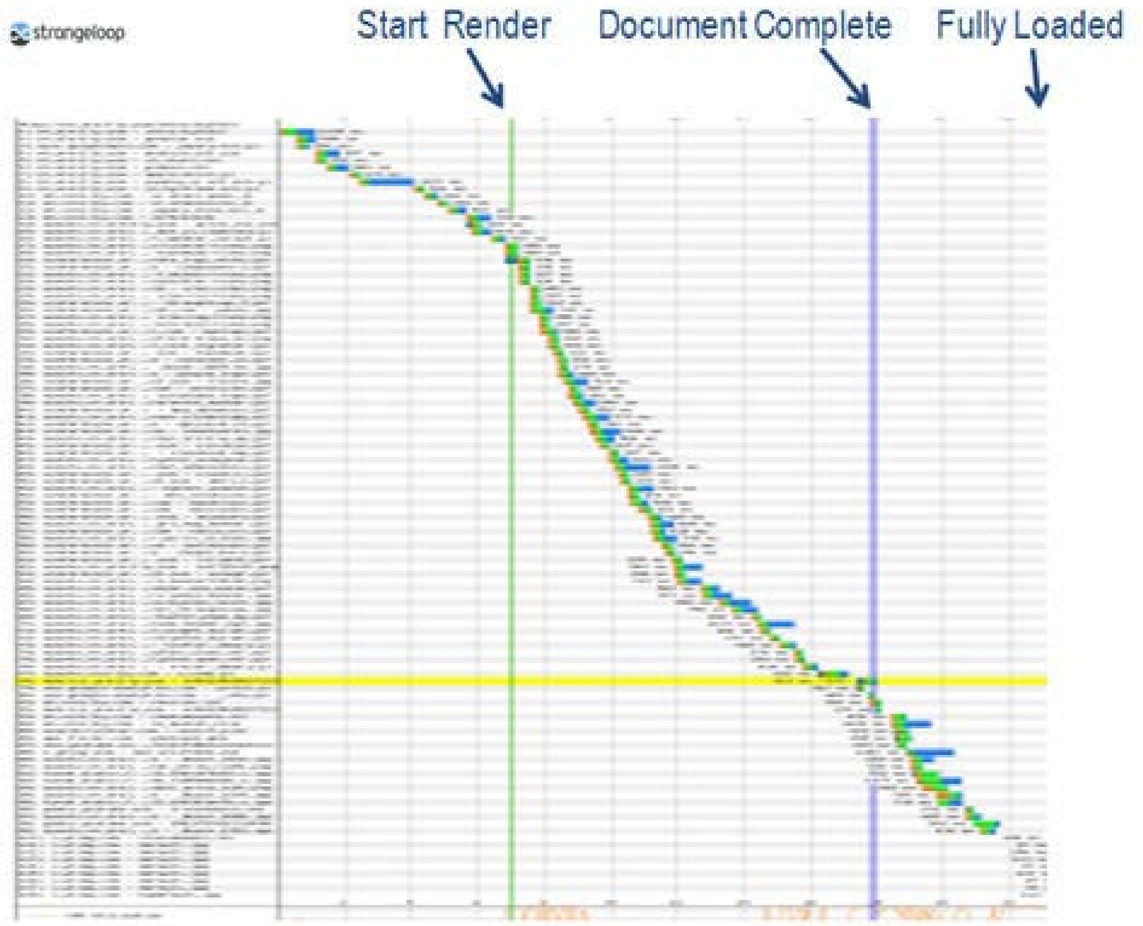
The user's internet connection is a factor here, but there are other things that can slow things down: the amount of time it takes your servers to "think" of what content to send, and the distance between your servers and the user.

Blue = Content download

This is the time it takes for every piece of content to be sent from the server to the browser. Think of this as the time it takes from the first sound you hear on the radio to the end of the speech from your mother after she broke in on your chat and told you to get off that walky-talky and come down for dinner.

You can speed up content download by making the total amount of stuff you need to send smaller. More on that in a minute.

For now, let's go back to our mondo chart:



I want you to observe four things:

1. The number of orange bars
2. The number of bright green bars
3. And the length of the bright green bars
4. The number of blue bars

These four things show where this site performs poorly. Let’s break it down:

1. Too many orange bars

You can address this problem by having your developers use something called “keep-alives” to reduce the number of TCP connections. (I’m not going to go into keep-alives here, because this post is long enough already. If you want more info, feel free to email me or ask in the comments.)

2. Too many bright green bars

Remember the 75 page objects I mentioned earlier, each one making its own roundtrip? That’s what’s causing all the green and blue bars. This problem can be fixed by implementing performance best practices, such as those identified by YSlow and Google. Among other things, these best practices include consolidating page objects into fewer bundles.

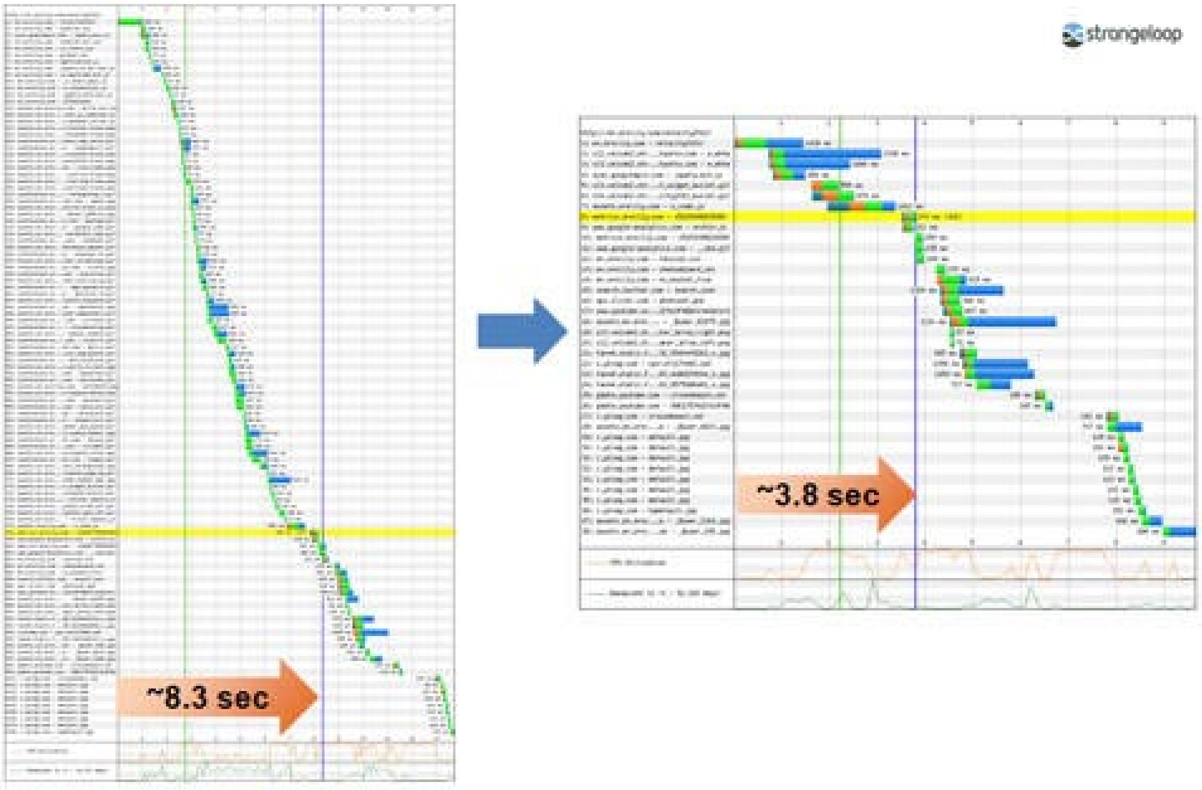
3. Bright green bars are too big

You can fix this problem with a CDN, which will bring your content closer to your users. Chances are, you’re already doing this. Hopefully, this post will help show why CDNs – while a great tool in your toolset – don’t address all aspects of the performance problem.

4. Too many blue bars

Not only are there too many page objects, each of these objects is too large. The average web page is between 320 and 420 kb. In other words, too big. This problem can be partially fixed through implementing the performance best practices suggested above, and partially fixed using something called compression. (Again, contact me if you want to know more about compression.)

So how does a waterfall chart look for a site that has been optimized to follow some of the best practices?** Take a look at these before-and-after charts for the same site:



If you take nothing else away from all this…

Remember that when you're looking at performance reports for your site, you want to see these five things:

1. As few rows as possible.
2. As few orange bars as possible.
3. Bright green bars that are as few and as short as possible.
4. As little blue as possible.
5. The "start render" and "document complete" vertical lines to occur as early as possible, and be as close together as possible.

*These charts were done by testing unoptimized and optimized versions of the Velocity 2010 website.
**Site optimization was done using Strangeloop's cloud-based Site Optimizer service.

By: Joshua Bixby