




A Client-Side Argument for Changing TCP Slow Start

Mike Belshe - mbelshe@chromium.org - Jan 11, 2010




A decorative header at the top of the slide features four overlapping spheres. From left to right, they are light green, light blue, light red, and light yellow. The spheres are partially cut off by the top edge of the slide.

Slow Start is a key part of
TCP's congestion control
algorithms...


<http://www.faqs.org/rfcs/rfc2581.html>



A decorative header featuring several overlapping circles in shades of green, blue, red, and yellow. A thin black horizontal line is positioned below the circles.

And, *"lack of attention to the dynamics of packet forwarding can result in severe service degradation or "Internet meltdown"*.

- Sally Floyd, <http://www.rfc-editor.org/rfc/rfc2914.txt>

A decorative header featuring four overlapping spheres: a green one on the left, and blue, red, and yellow ones on the right, all partially cut off by the top edge of the slide.

But Web Browsers today are
inadvertently subverting slow
start....



A decorative header featuring four overlapping spheres: a green one on the left, and blue, red, and yellow ones on the right. A thin black horizontal line is positioned below the spheres.

HTTP Background



HTTP originally was simple...

1. Open a connection
2. Send a request
3. Receive a response
4. When the server hangs up, you're done!

Couldn't be easier to implement...

But not very efficient on the network...



But Inefficiency Didn't Matter Much

- Web Pages were tiny
- Just a little text and a picture of a dog or two
- No CSS, JS, Video, etc.

As Web Pages grew, simple optimizations were built:

- Keep-Alives recycle TCP connections
- Better caching support
- etc

But....



Web Pages Continued to Grow

Technologies evolved.

- CSS, JavaScript, and image usage ballooned
- The average site in the Top-100 now makes ~40 requests per page (uncached)

Browsers were still limited to 2 connections per site (per spec).

Workarounds were easy...



Circumventing Slow Start

- Sending only two concurrent requests was a bottleneck
- Sites seeking speed needed more connections
 - "Subdomain sharding" was invented, since existing browsers would open 2 connections to each subdomain
- Eventually browsers kicked up connection limits
 - 2 connections per domain became 6

And voila! Slow Start is subverted.

6 Connections Per Domain

* 6 Subdomains

* 3 initial cwnd per connection

108 initial cwnd (not exactly what RFC2581 suggests!)



Who is Wrong?

Is the TCP specification wrong?

- Is Slow Start too aggressive?
- Maybe the internet won't break if you turn it off...

Or maybe this is why we measure 1-1.5% packet loss on web pages today...

I don't know, but these are the facts.



Trying to Build a Better Protocol

SPDY attempts to solve some of HTTP's more fundamental problems:

- serialized requests
- not enough compression

And it is more efficient:

- Overall 40% reduction in packets
- 15% reduction in bytes transmitted

And *most* of the time it is faster.



Slow Start Gets In The Way

SPDY attempts to be efficient by using the network more intelligently.

Fewer TCP Connections.

But

- SPDY uses a single connection per domain.
- HTTP uses 6 connections per domain.

This gives HTTP an initial cwnd **6 times larger** than an "efficient" protocol!



A decorative header at the top of the slide features four overlapping spheres. From left to right, they are light green, light blue, light red, and light yellow. The spheres are partially cut off by the top edge of the slide.

Example: The Slow Start Bottleneck in Action

In This Example

Network:

- 100Mbps
- 200ms RTT
- 0% loss

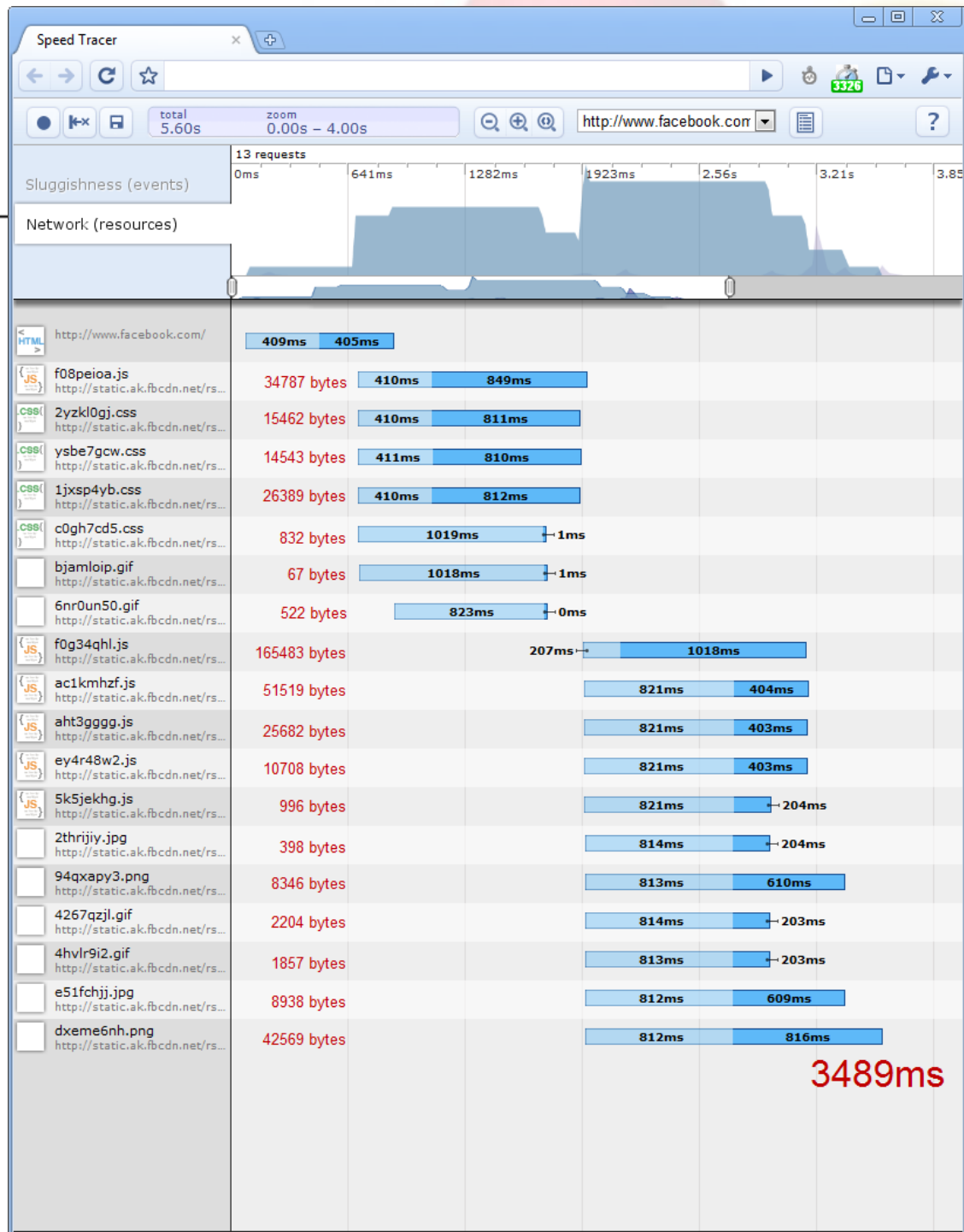
Browser

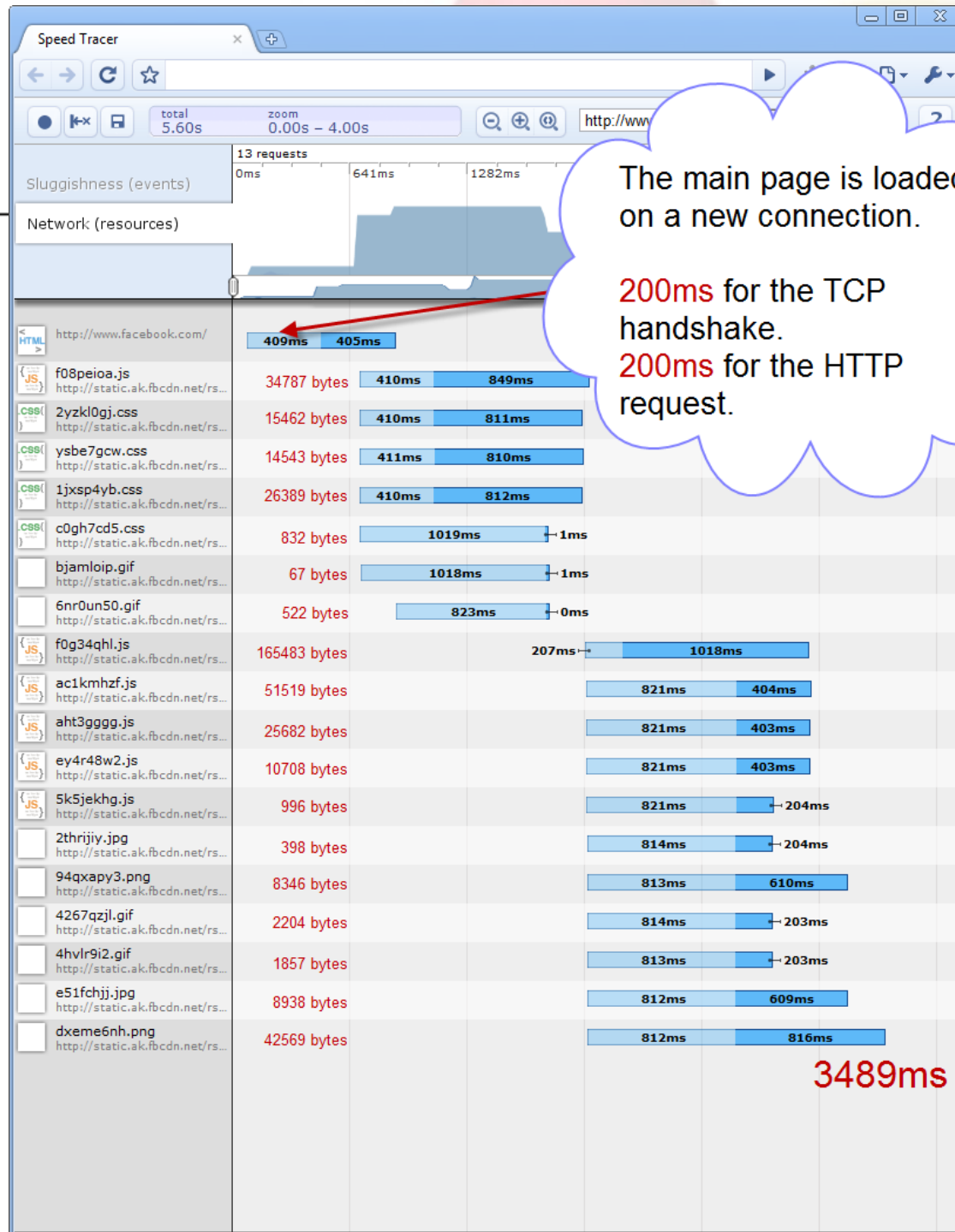
- Chrome
- Using SPDY, to load the page over a single connection
- No resources are cached

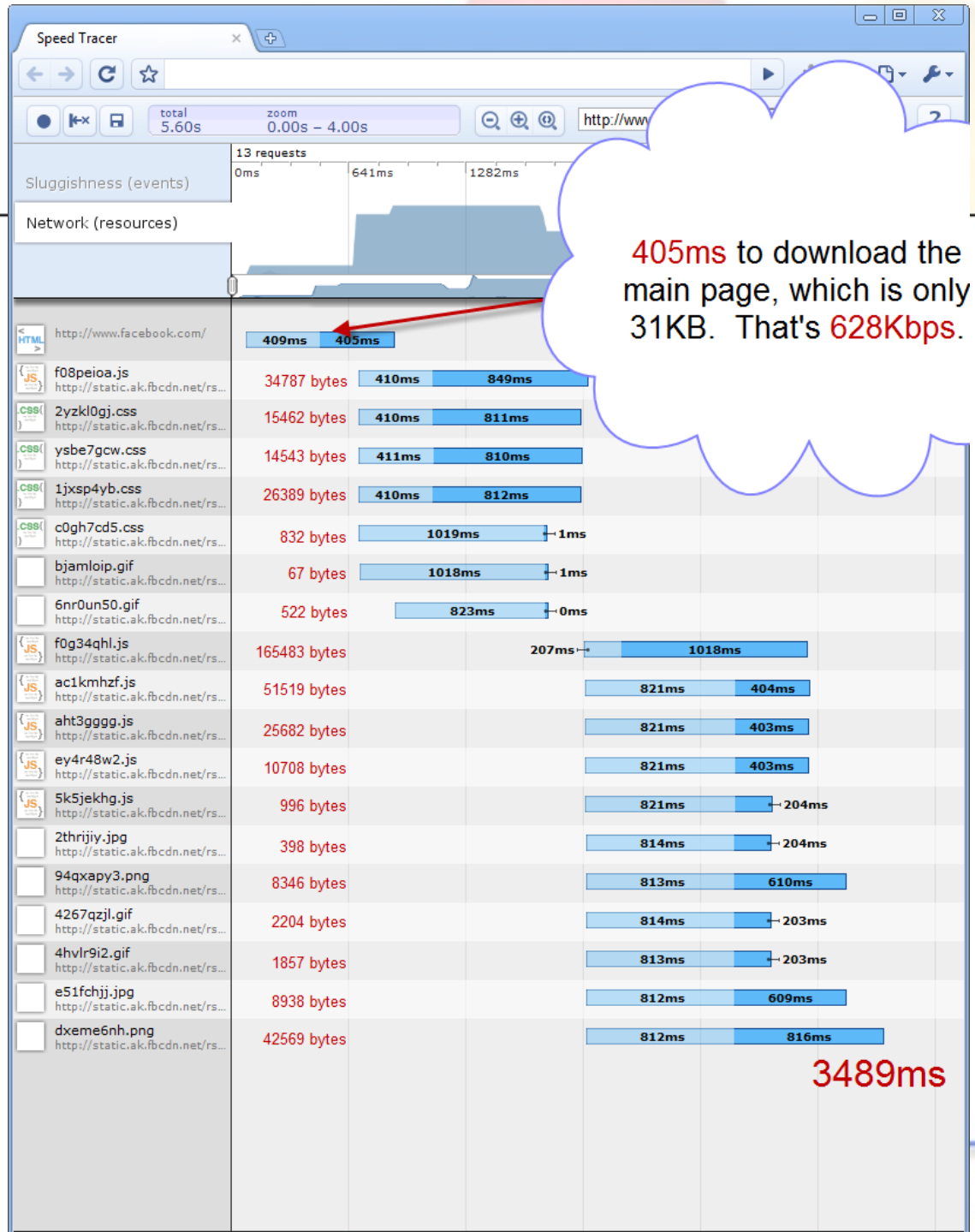
WebPage

- <http://www.facebook.com/>
- But this can be witnessed on many many webpages.
- There is nothing wrong with the facebook page content.





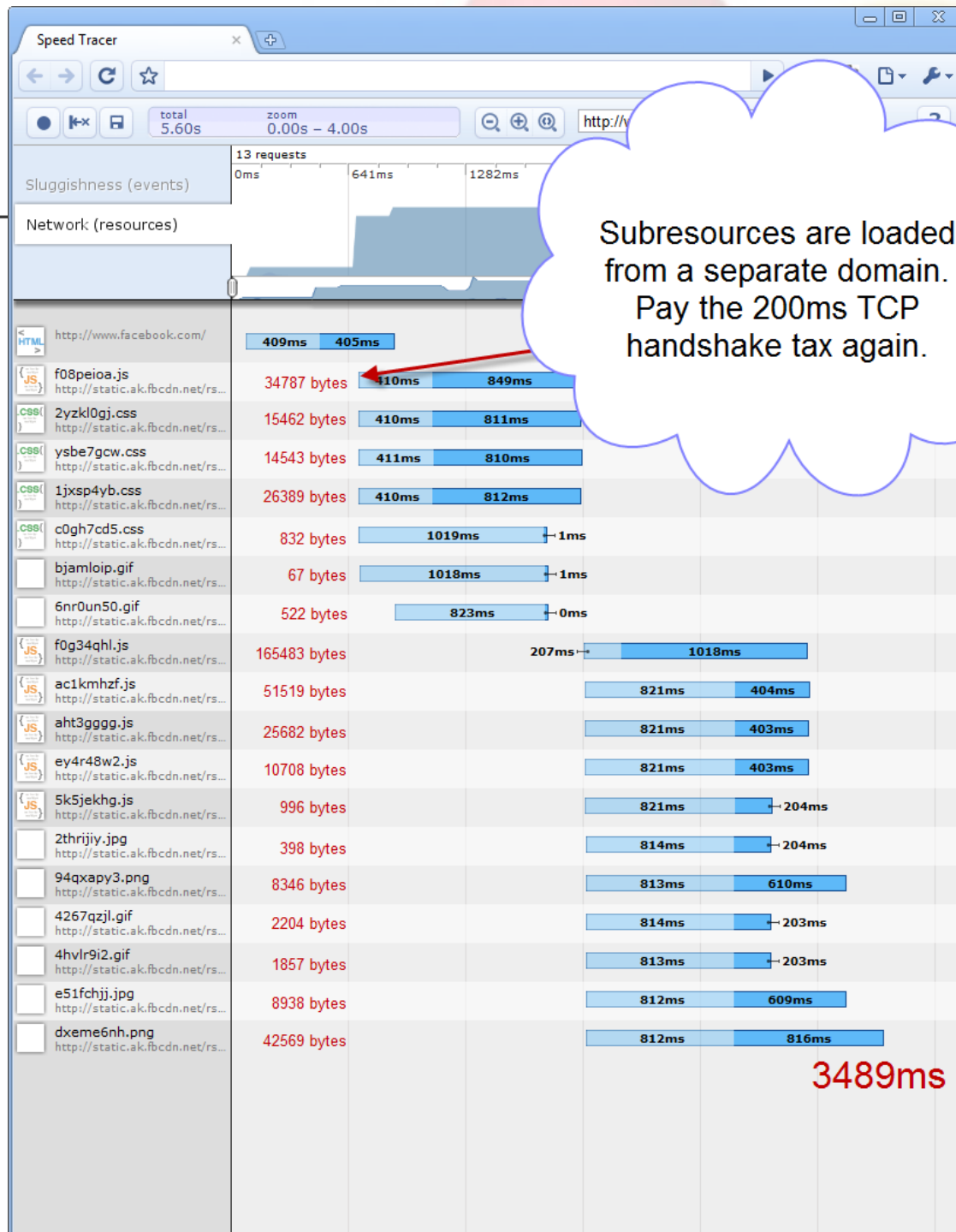


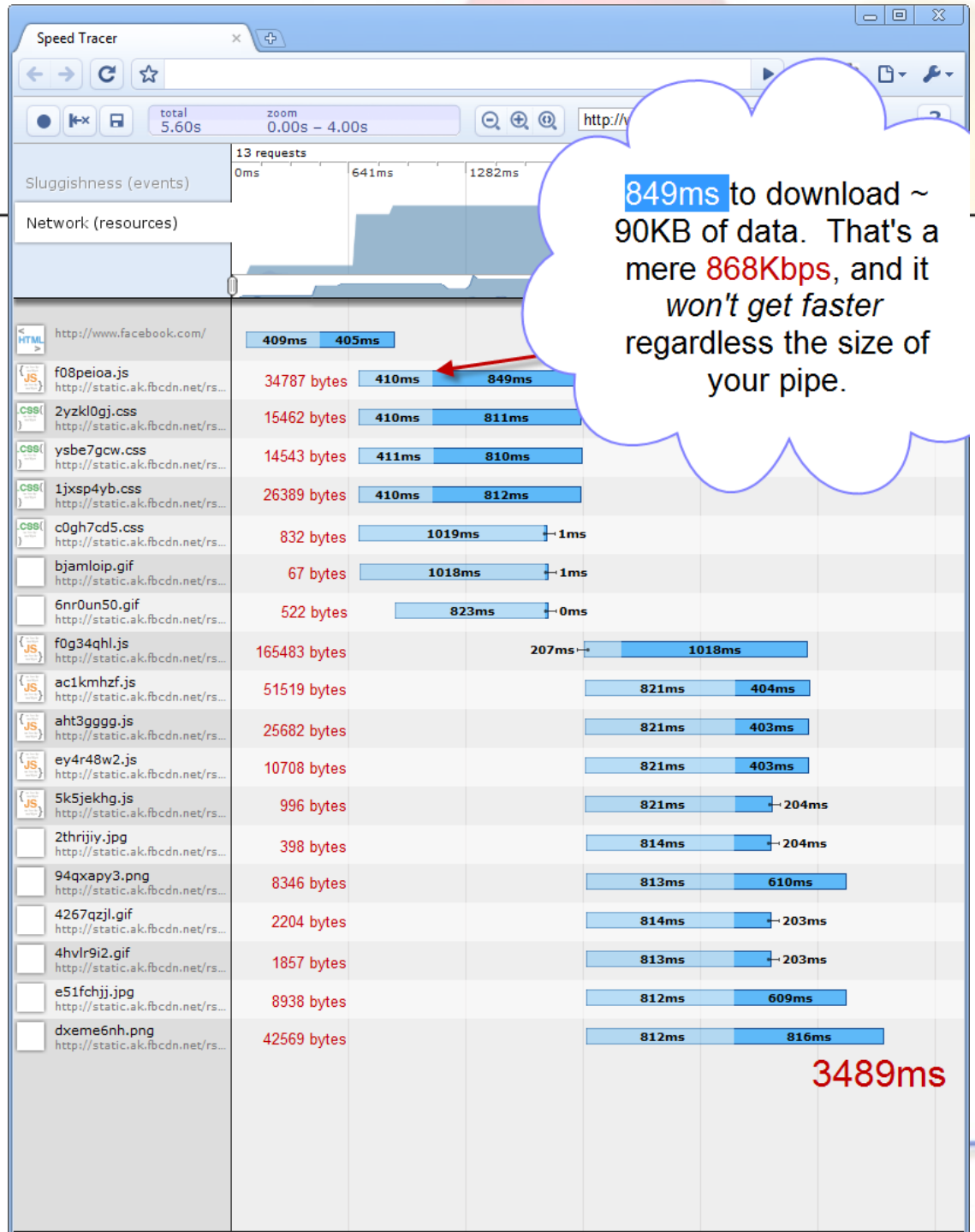


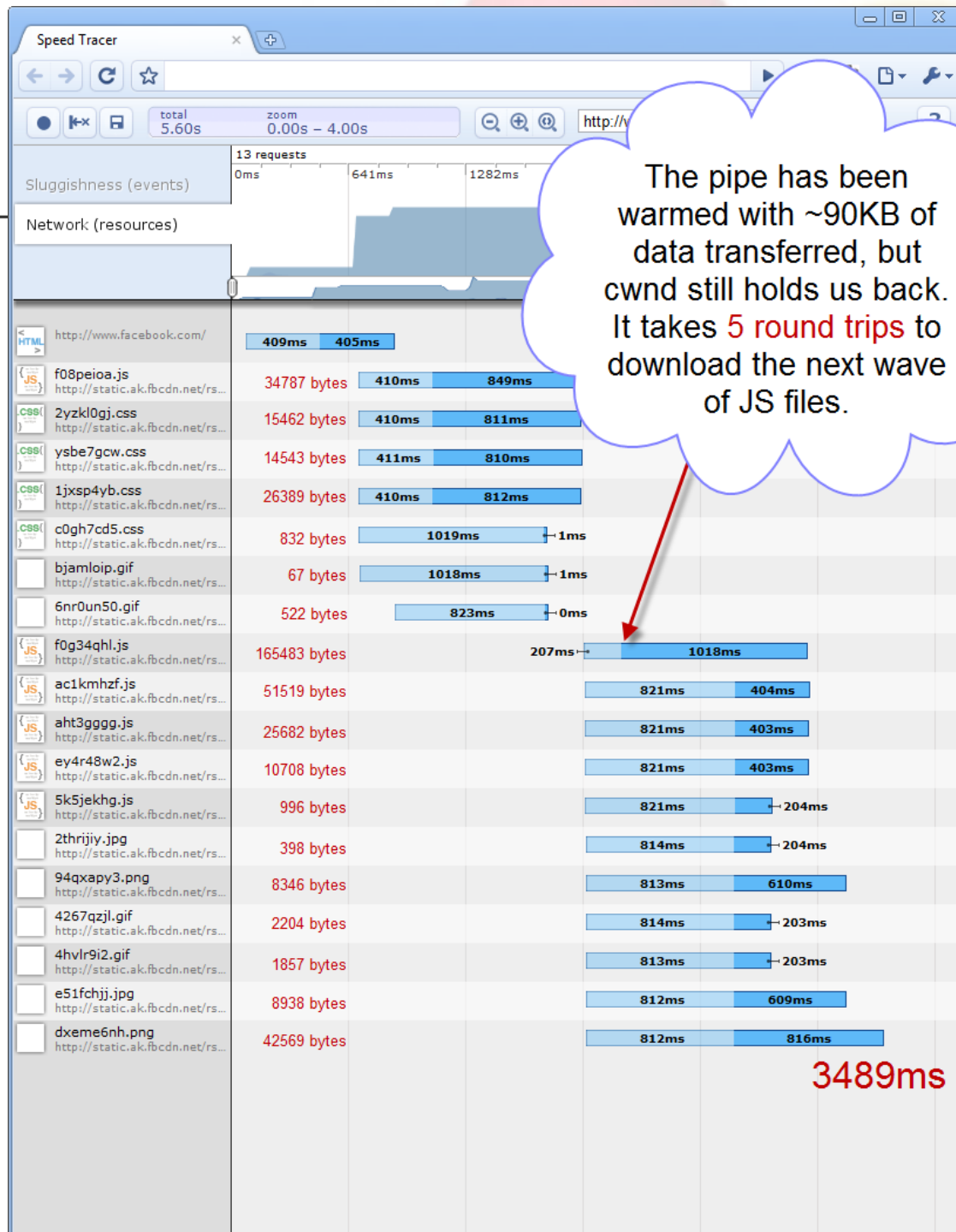
405ms to download the main page, which is only 31KB. That's 628Kbps.

3489ms



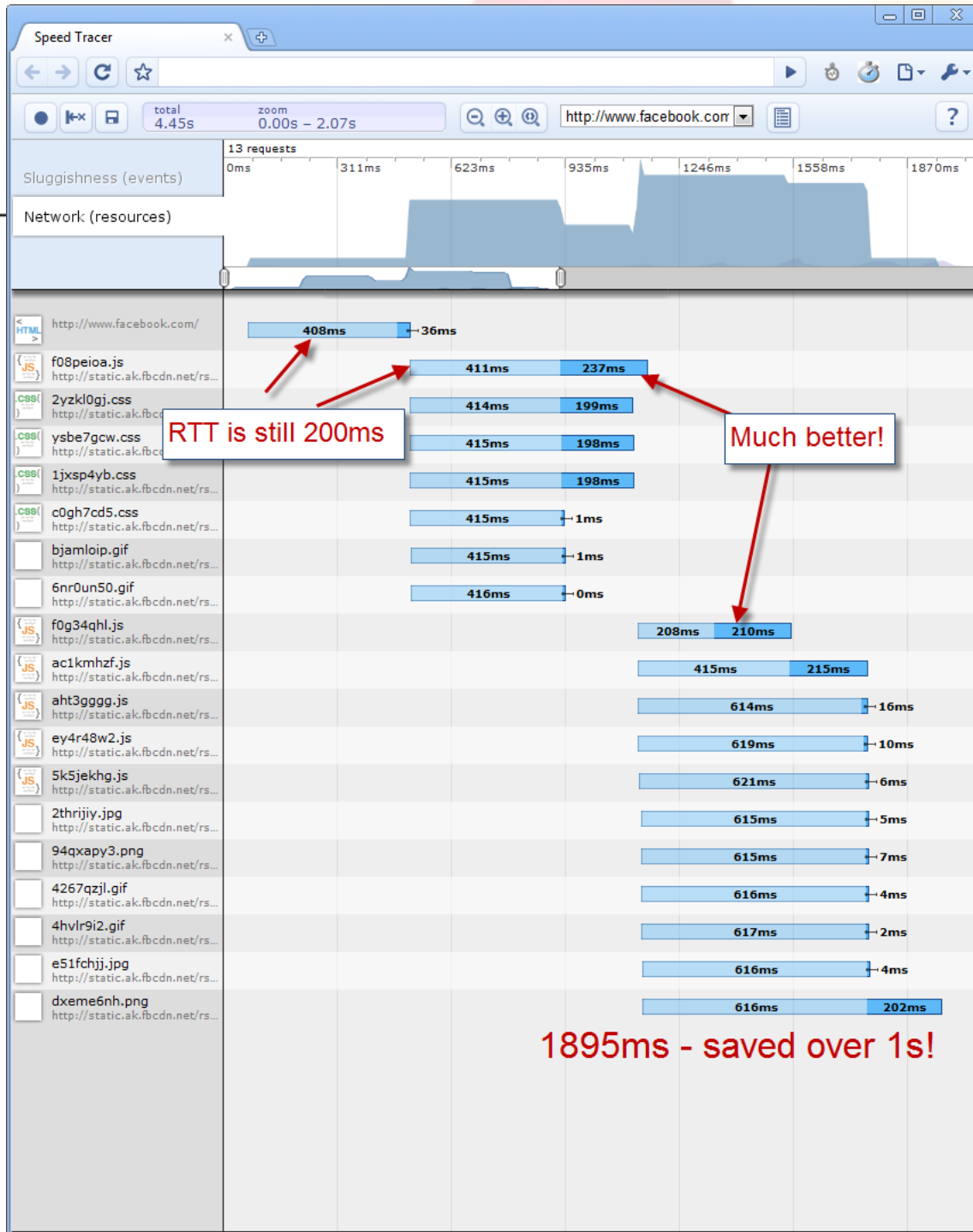






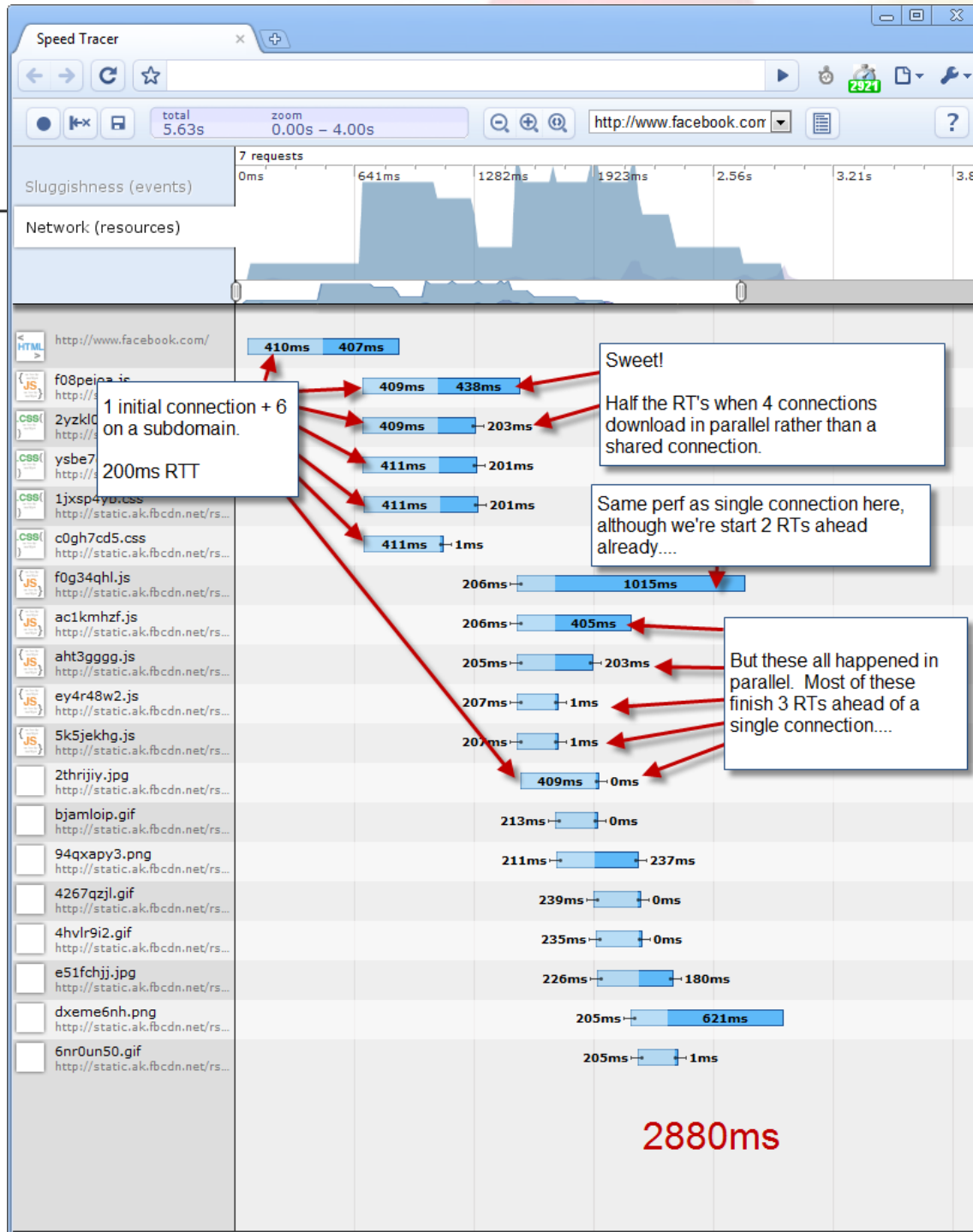
A decorative header at the top of the slide features four overlapping spheres: a green one on the far left, a blue one in the middle, a red one partially behind the blue one, and a yellow one on the right. A thin black horizontal line runs across the slide just below these spheres.

What happens if we increase
the initial cwnd to..... 18?



A decorative header at the top of the slide features four overlapping spheres. From left to right, they are light green, light blue, light red, and light yellow. The spheres are partially cut off by the top edge of the slide.

Compare to Traditional HTTP



Increasing Initial CWND?

It's not a matter of whether we should increase initial cwnd. *For HTTP, it is already increased.*

Data:

Sites are already using $\text{init-cwnd} = 3$.

HTTP uses 6 connections per domain, enabling $\text{init-cwnd} = 18$.

Sites use 3-8 subdomains, increasing init-cwnd to 54-144.



But Is It Realistic?

initcwnd is generally a global setting hidden inside TCP.

If we increase it globally, HTTP's init-cwnd is always at least 6X larger than it should be.

Even with SPDY, until sites stop using subdomain sharding, effective cwnd is higher than what TCP is attempting to enforce.

Will the internet at large increase their cwnd? Needs a kernel patch at the minimum - possibly a sockets API change.



Conclusions

- Any TCP-based protocol attempting to be maximally efficient on the network will use fewer TCP connections than HTTP.
- But using fewer connections is a handicap due to TCP's Slow Start on high latency links.
- Slow Start, while specified at ~2-4 pkts, has, in practice, already been increased far beyond that by use of multiple connections and subdomains.
- If TCP cannot increase initcwnd, the next generation HTTP protocol will likely not be TCP based.