SPDY and What to Consider for HTTP/2.0

mike belshe

Why am I here?

SPDY started over 3 years ago

Reduced latency is now proven

It's better for the network

Let's focus on interoperability

Who is using SPDY?

- Google Chrome &
  All Google Web Properties
- Mozilla Firefox
- Twitter
- Amazon Silk
- Others: Cotendo, Strangeloop, iPhone client, Apache mod-spdy, nginx beta, jetty, netty, libraries in python, node.js, erlang, ruby, go, and C

How did SPDY come to be?

wanted reduced web page latency *for users*

What SPDY is Not

A transport layer protocol (like TCP)

Rocket Science

Cheap Compression Tricks

What SPDY is

An amalgam of well-known ideas based on performance data:

multiplexing
prioritization
compression
server push
transparent to HTTP app servers
deployable today

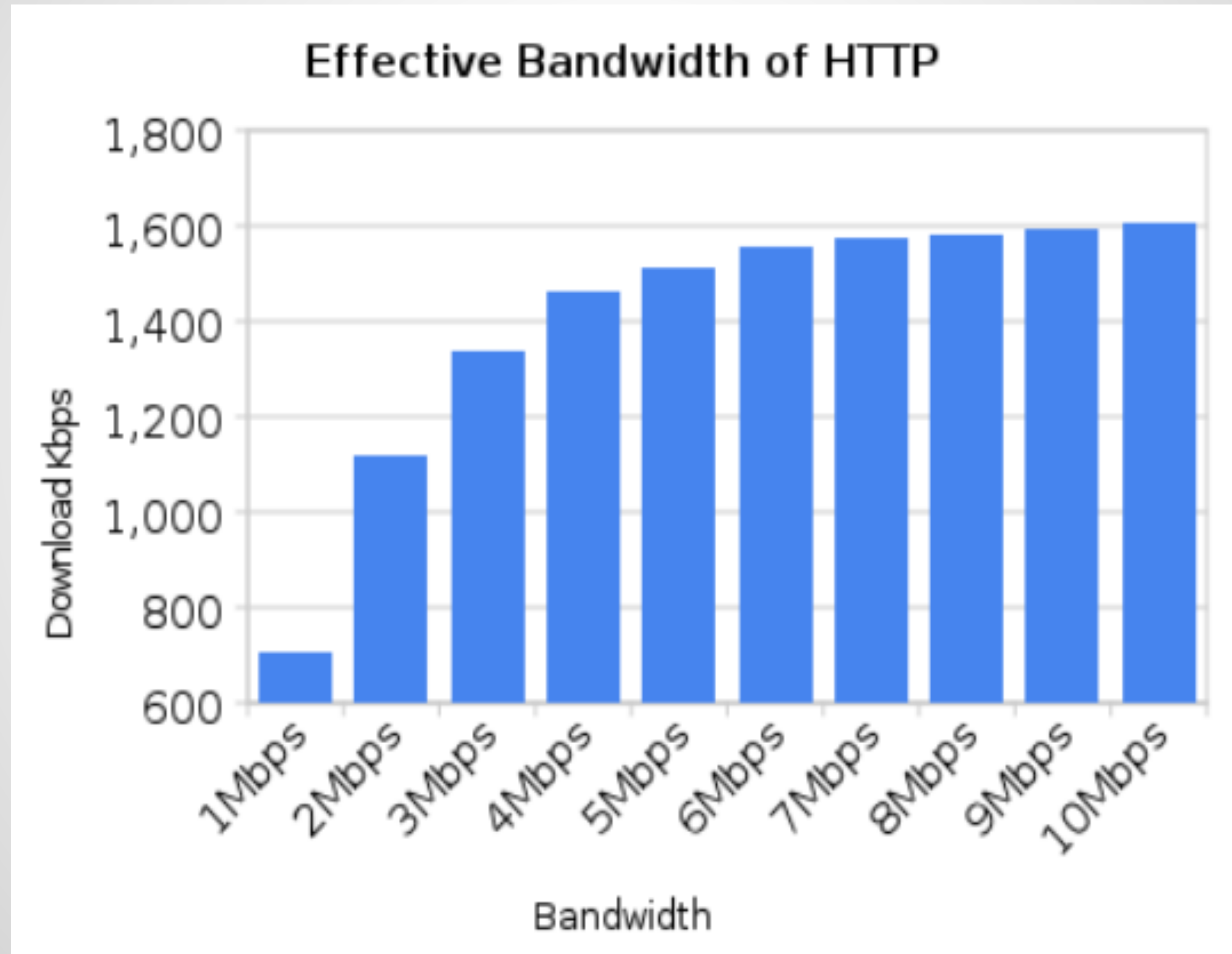
Real deployment has shown also

- Better for the network
- Better for Mobile
  HTTP is not just for HTML
  Battery life matters

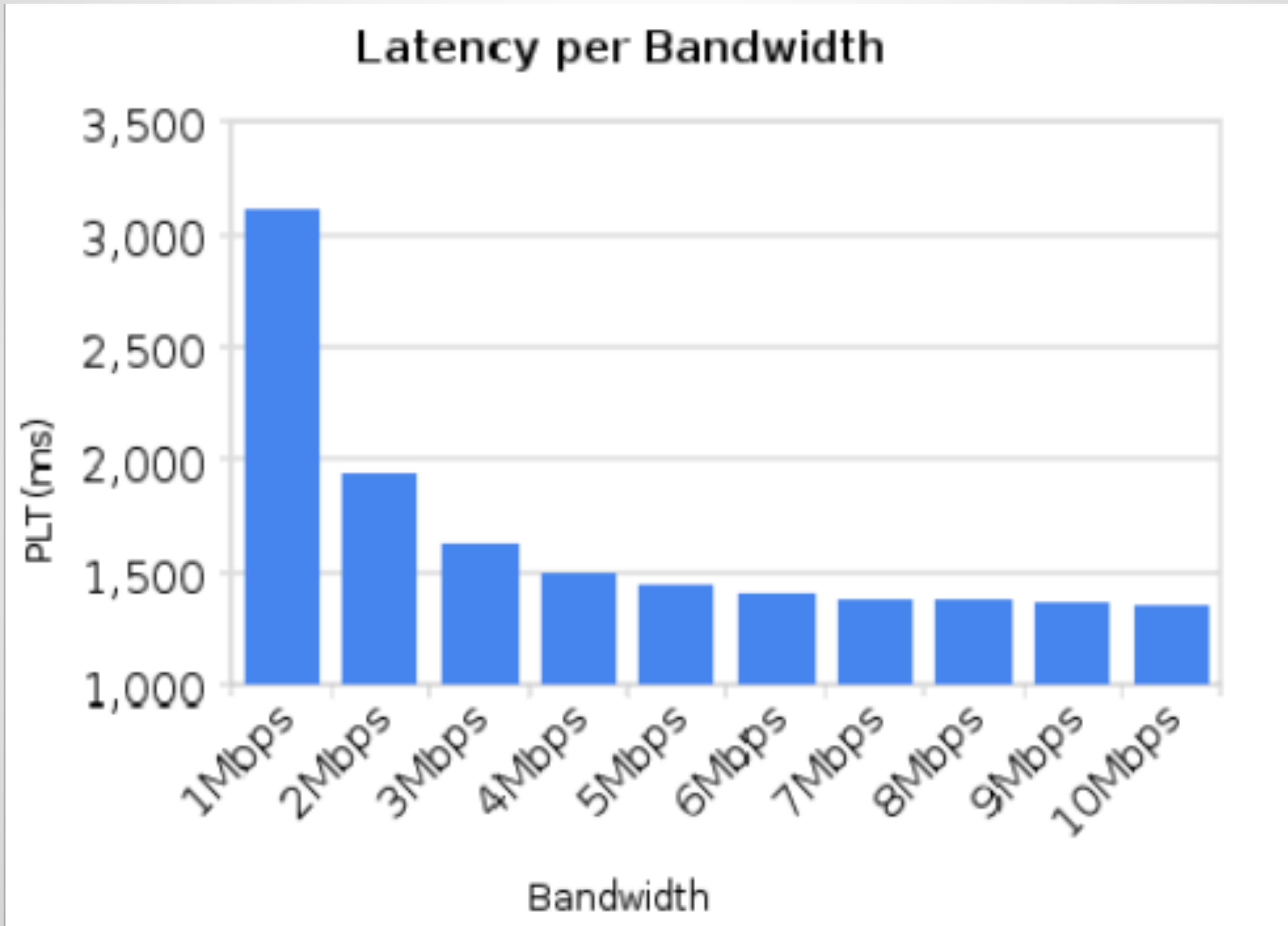
Background: What is a WebPage?

- 86 resources
- 13 hosts
- 800+KB
- only 66% compressed (top sites are ~90% compressed)

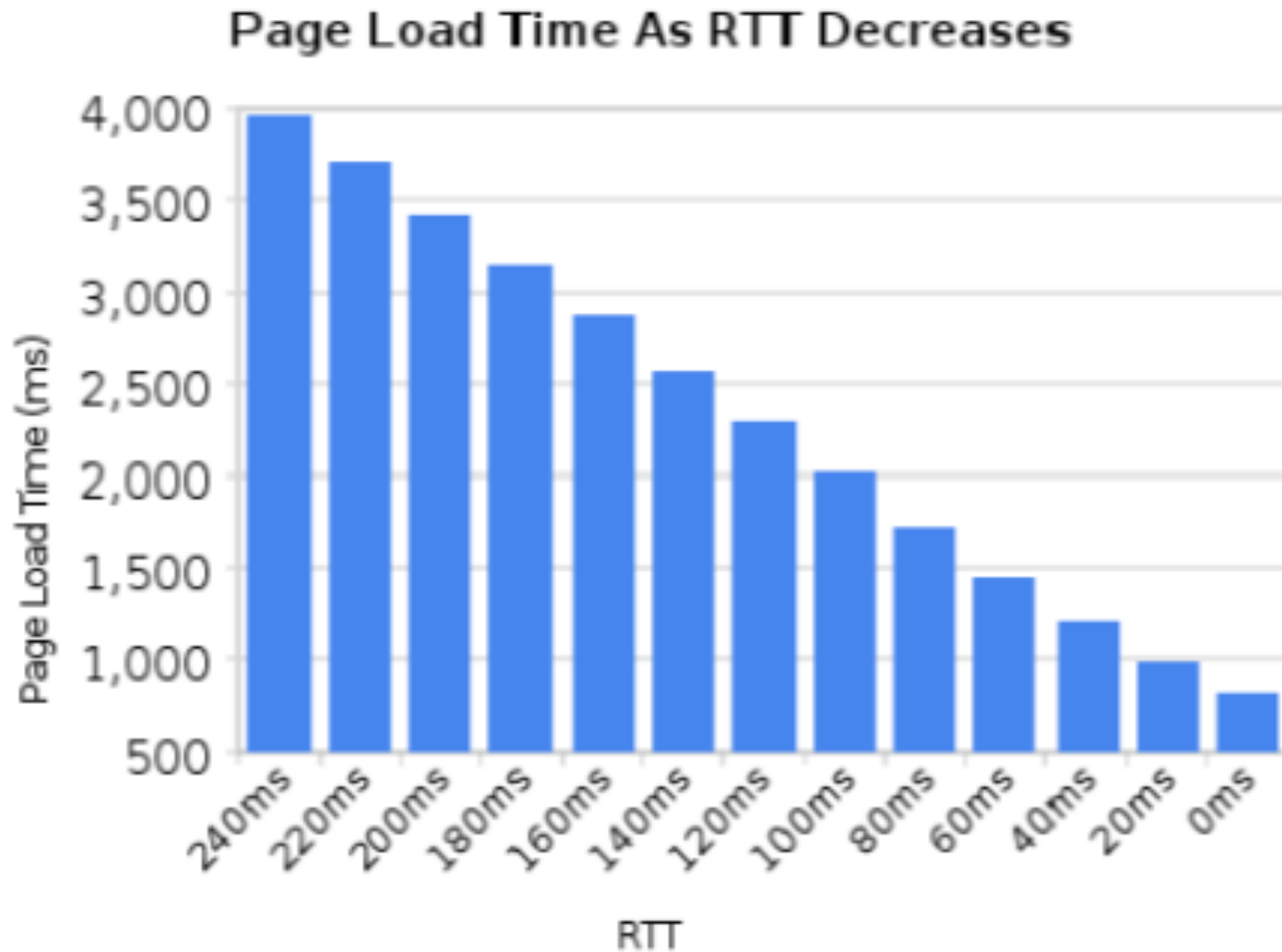
Background: Poor Network Utilization



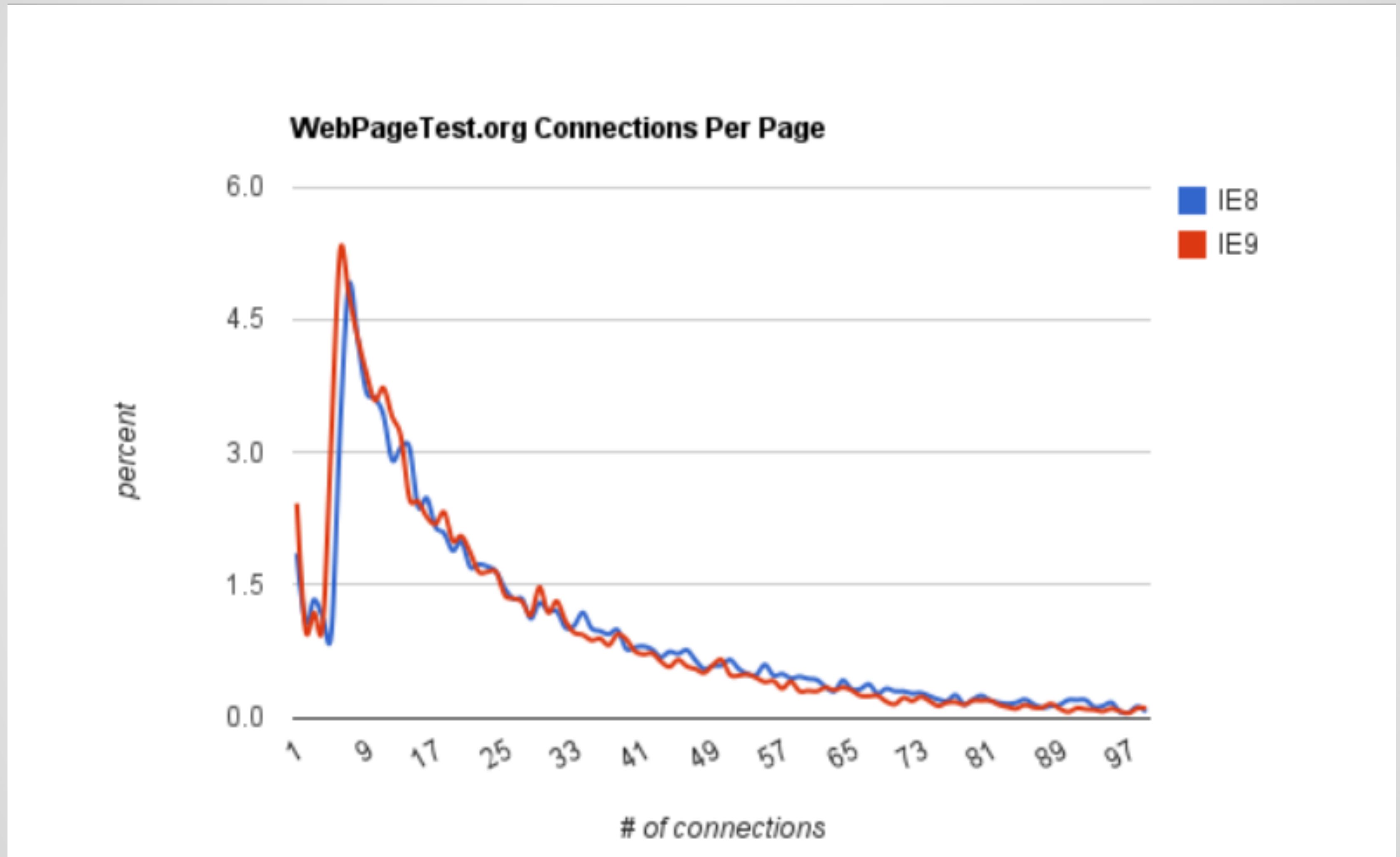
More Bandwidth Doesn't Help



But Reducing Round Trip Time Does



Background: HTTP Connections



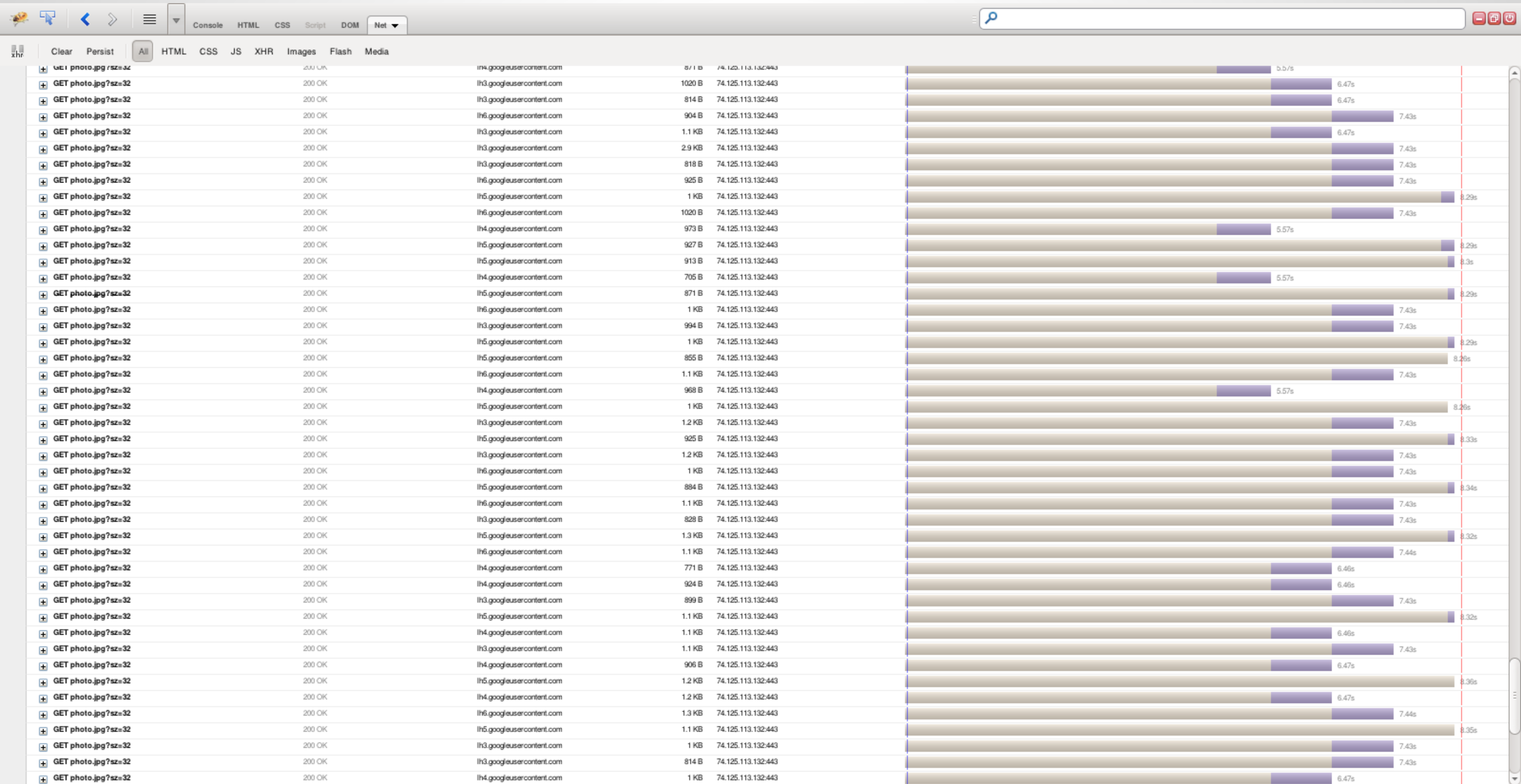
2010: Average 29 connections per page.

SPDY Features

1. Multiplexing

- Small, fixed length frames
- Fully interleaved streams
- Streams can be created by either endpoint with zero round trips.
- Many implementors have remarked it's easy to implement!

Before Multiplexing



After Multiplexing

Console HTML CSS Script DOM Net

Clear Persist All HTML CSS JS XHR Images Flash Media

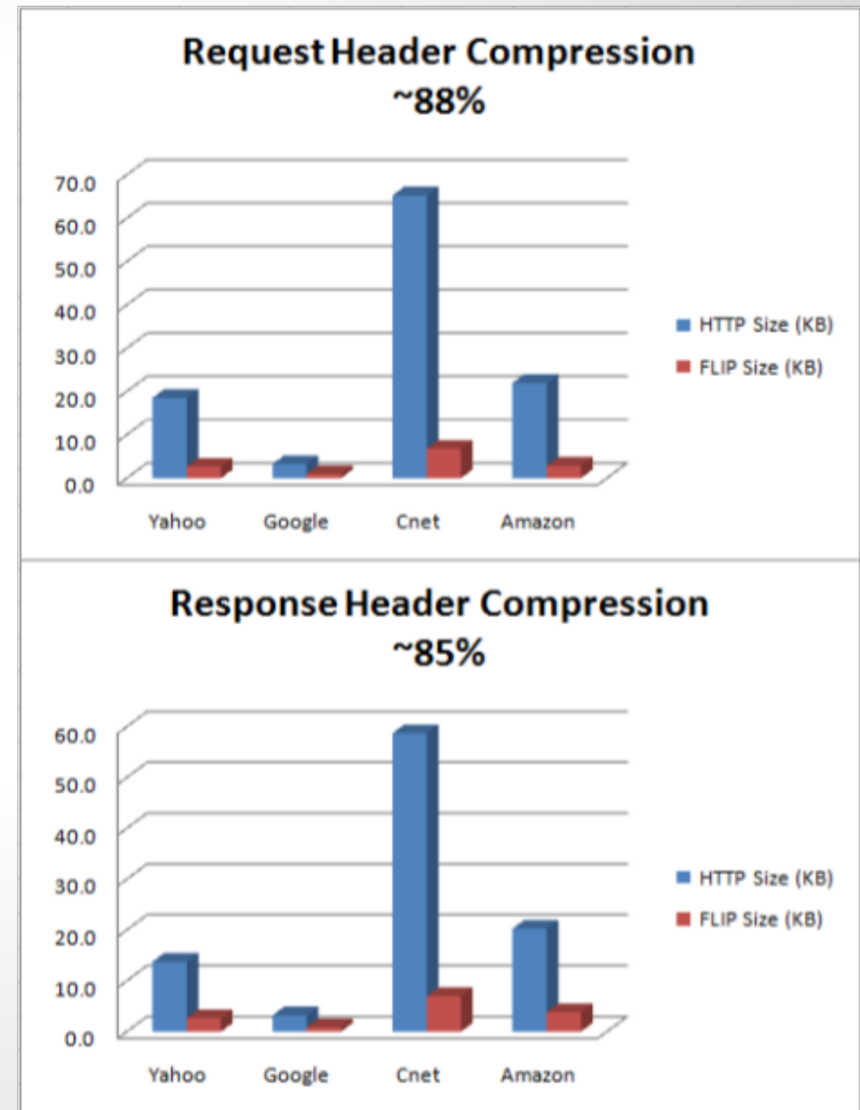
Request	Status	Domain	Size	Remote IP	Time
GET photo.jpg?sz=32	200 OK	lh3.googleusercontent.com	1.2 KB	74.125.113.132:443	150ms
GET photo.jpg?sz=32	200 OK	lh3.googleusercontent.com	870 B	74.125.113.132:443	150ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	907 B	74.125.113.132:443	149ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	991 B	74.125.113.132:443	149ms
GET photo.jpg?sz=32	200 OK	lh3.googleusercontent.com	1 KB	74.125.113.132:443	149ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	924 B	74.125.113.132:443	150ms
GET photo.jpg?sz=32	200 OK	lh4.googleusercontent.com	1.1 KB	74.125.113.132:443	150ms
GET photo.jpg?sz=32	200 OK	lh4.googleusercontent.com	942 B	74.125.113.132:443	150ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	1 KB	74.125.113.132:443	150ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	1.4 KB	74.125.113.132:443	151ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	2.9 KB	74.125.113.132:443	151ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	1 KB	74.125.113.132:443	152ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	965 B	74.125.113.132:443	152ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	1.1 KB	74.125.113.132:443	152ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	847 B	74.125.113.132:443	153ms
GET photo.jpg?sz=32	200 OK	lh4.googleusercontent.com	1 KB	74.125.113.132:443	153ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	1 KB	74.125.113.132:443	151ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	1 KB	74.125.113.132:443	152ms
GET photo.jpg?sz=32	200 OK	lh3.googleusercontent.com	835 B	74.125.113.132:443	152ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	1.1 KB	74.125.113.132:443	152ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	1000 B	74.125.113.132:443	152ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	2.6 KB	74.125.113.132:443	153ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	1.2 KB	74.125.113.132:443	153ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	975 B	74.125.113.132:443	154ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	970 B	74.125.113.132:443	153ms
GET photo.jpg?sz=32	200 OK	lh4.googleusercontent.com	1 KB	74.125.113.132:443	153ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	662 B	74.125.113.132:443	154ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	990 B	74.125.113.132:443	154ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	1 KB	74.125.113.132:443	155ms
GET photo.jpg?sz=32	200 OK	lh4.googleusercontent.com	2.7 KB	74.125.113.132:443	155ms
GET photo.jpg?sz=32	200 OK	lh3.googleusercontent.com	2.7 KB	74.125.113.132:443	155ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	906 B	74.125.113.132:443	156ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	2.8 KB	74.125.113.132:443	156ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	1.1 KB	74.125.113.132:443	157ms
GET photo.jpg?sz=32	200 OK	lh3.googleusercontent.com	964 B	74.125.113.132:443	158ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	2.8 KB	74.125.113.132:443	158ms
GET photo.jpg?sz=32	200 OK	lh4.googleusercontent.com	1.1 KB	74.125.113.132:443	159ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	1 KB	74.125.113.132:443	159ms
GET photo.jpg?sz=32	200 OK	lh3.googleusercontent.com	968 B	74.125.113.132:443	159ms
GET photo.jpg?sz=32	200 OK	lh4.googleusercontent.com	1.4 KB	74.125.113.132:443	159ms
GET photo.jpg?sz=32	200 OK	lh4.googleusercontent.com	954 B	74.125.113.132:443	160ms
GET photo.jpg?sz=32	200 OK	lh3.googleusercontent.com	1003 B	74.125.113.132:443	160ms
GET photo.jpg?sz=32	200 OK	lh5.googleusercontent.com	1.1 KB	74.125.113.132:443	160ms
GET photo.jpg?sz=32	200 OK	lh4.googleusercontent.com	1016 B	74.125.113.132:443	161ms
GET photo.jpg?sz=32	200 OK	lh6.googleusercontent.com	736 B	74.125.113.132:443	161ms

2. Prioritization

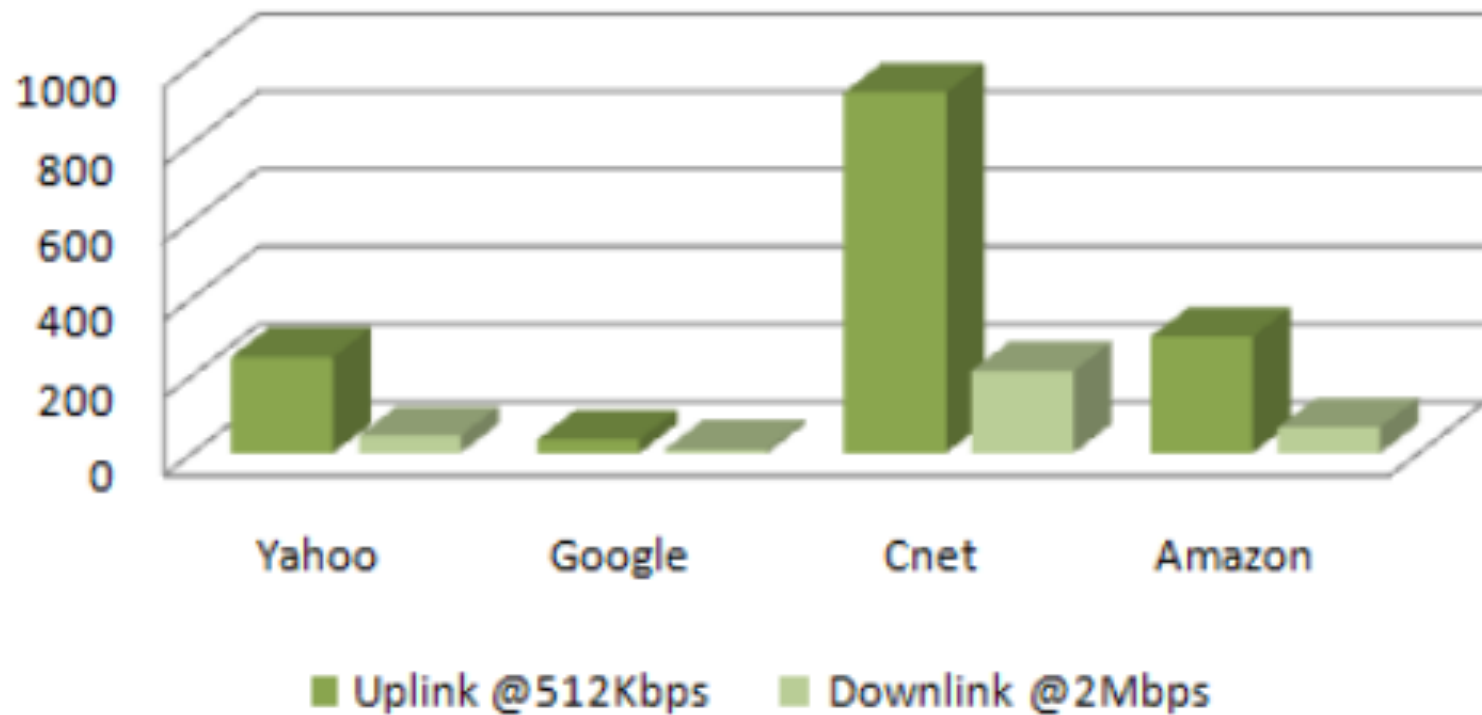
- Not all requests are equal!
- Failure to prioritize is actually slower
- Must consider two metrics:
  - Time to first render
  - Overall Page Load Time
- SPDY allows client-specified priorities with server best effort to deliver

3. Header Compression

- SPDY uses stateful compression across requests
- Using zlib, achieves 85-90% compression
- Don't care if compressor is zlib; only care about session state.
- Must be mandatory

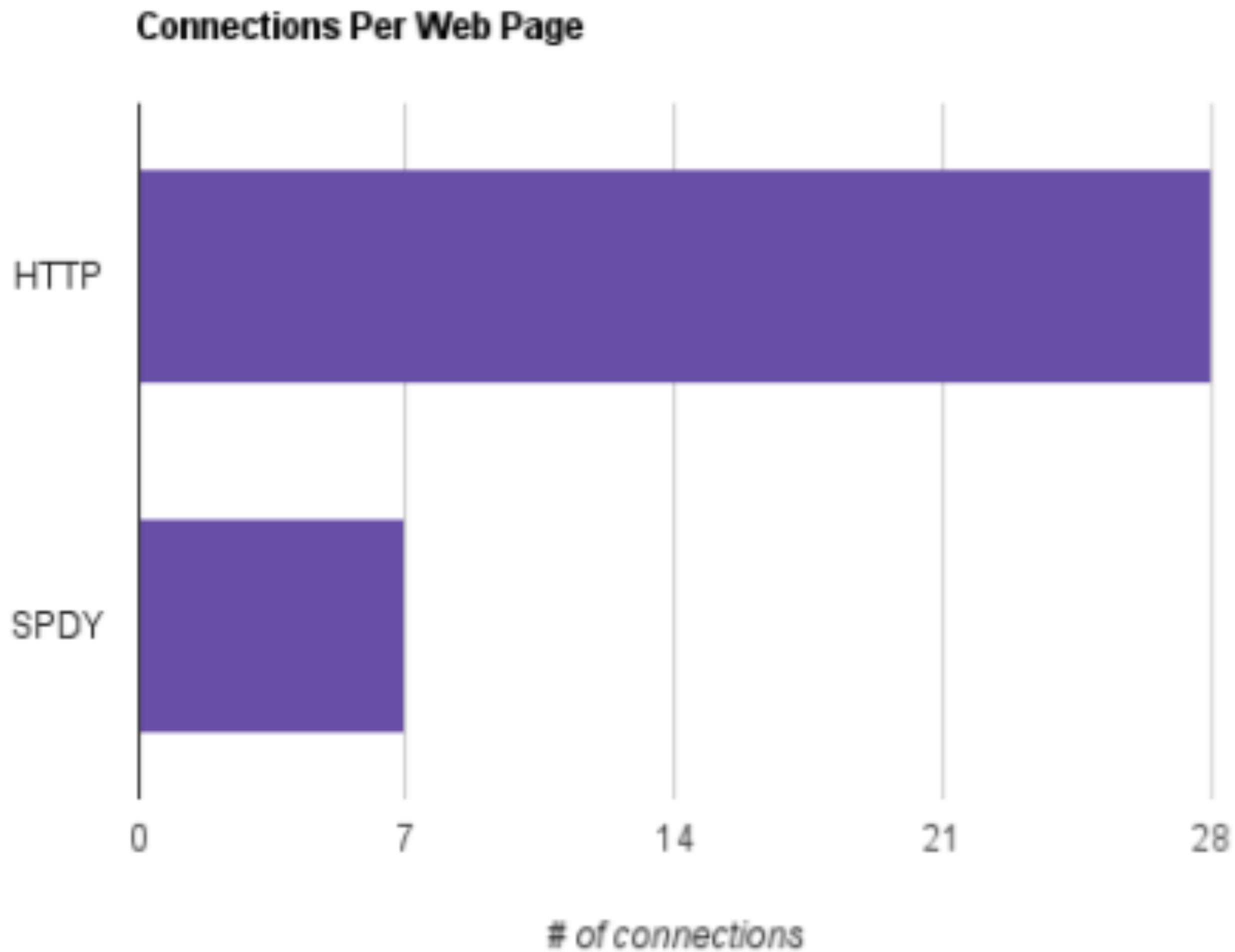


Compression Savings

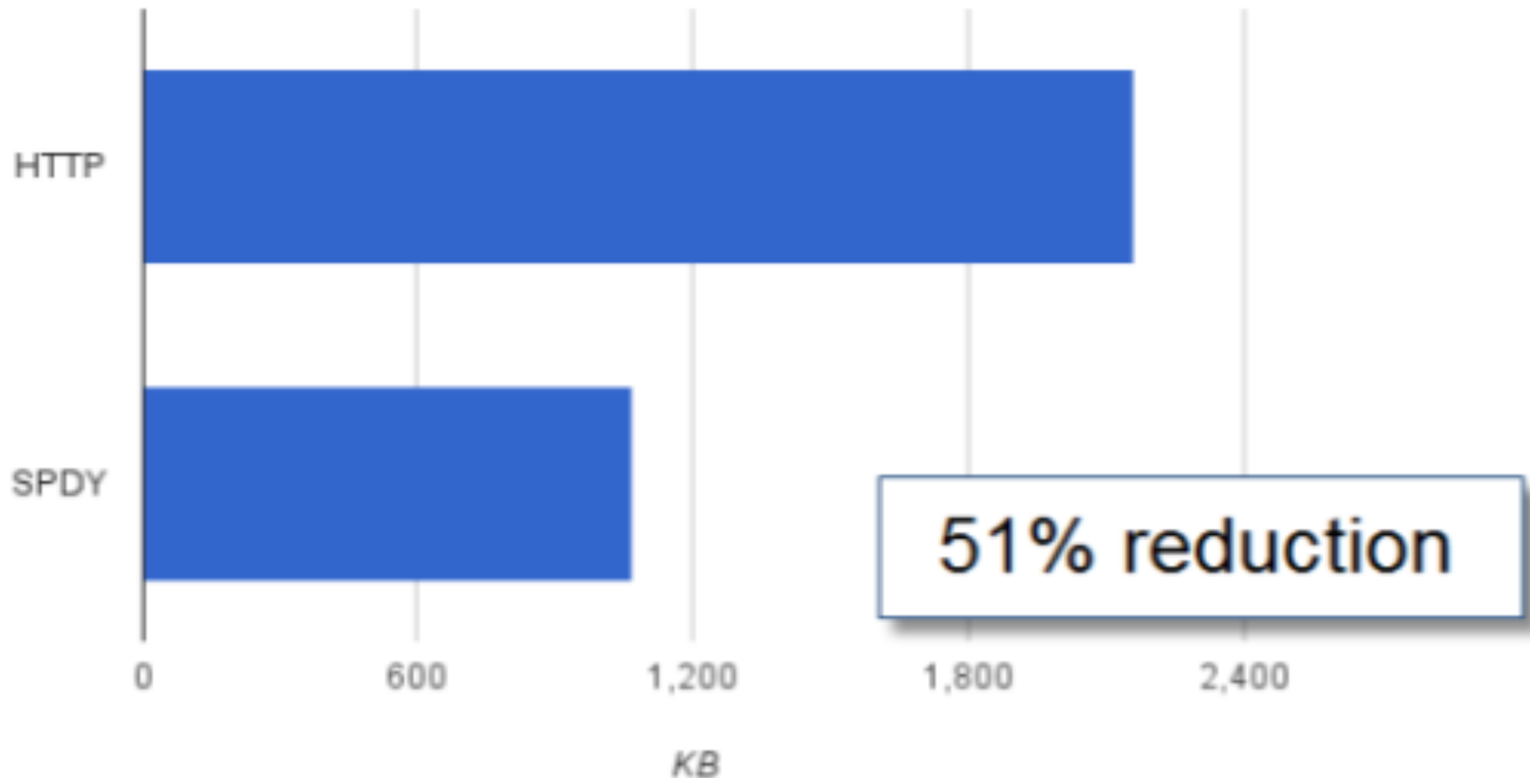


Better Networking

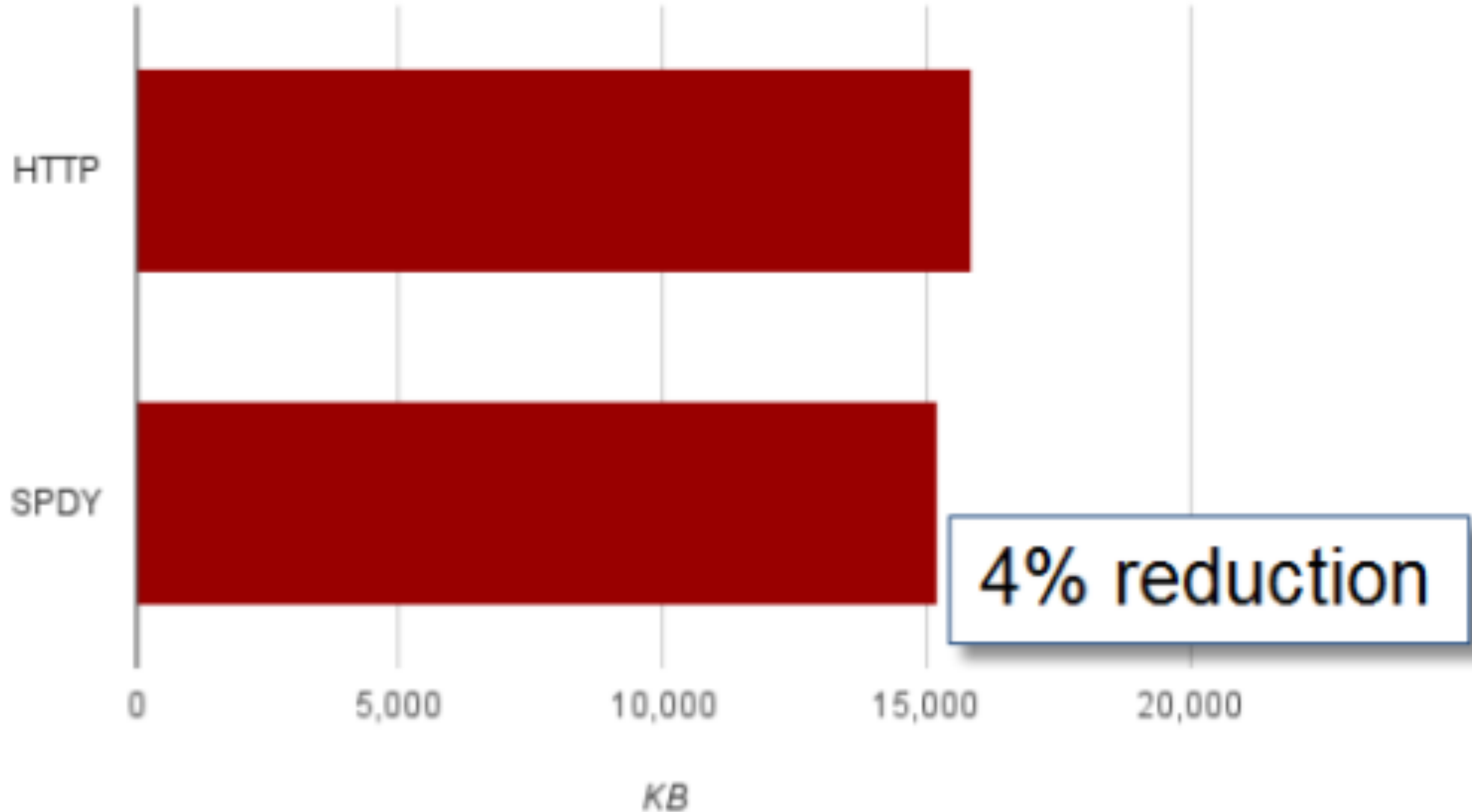
SPDY "Less is More" Connections



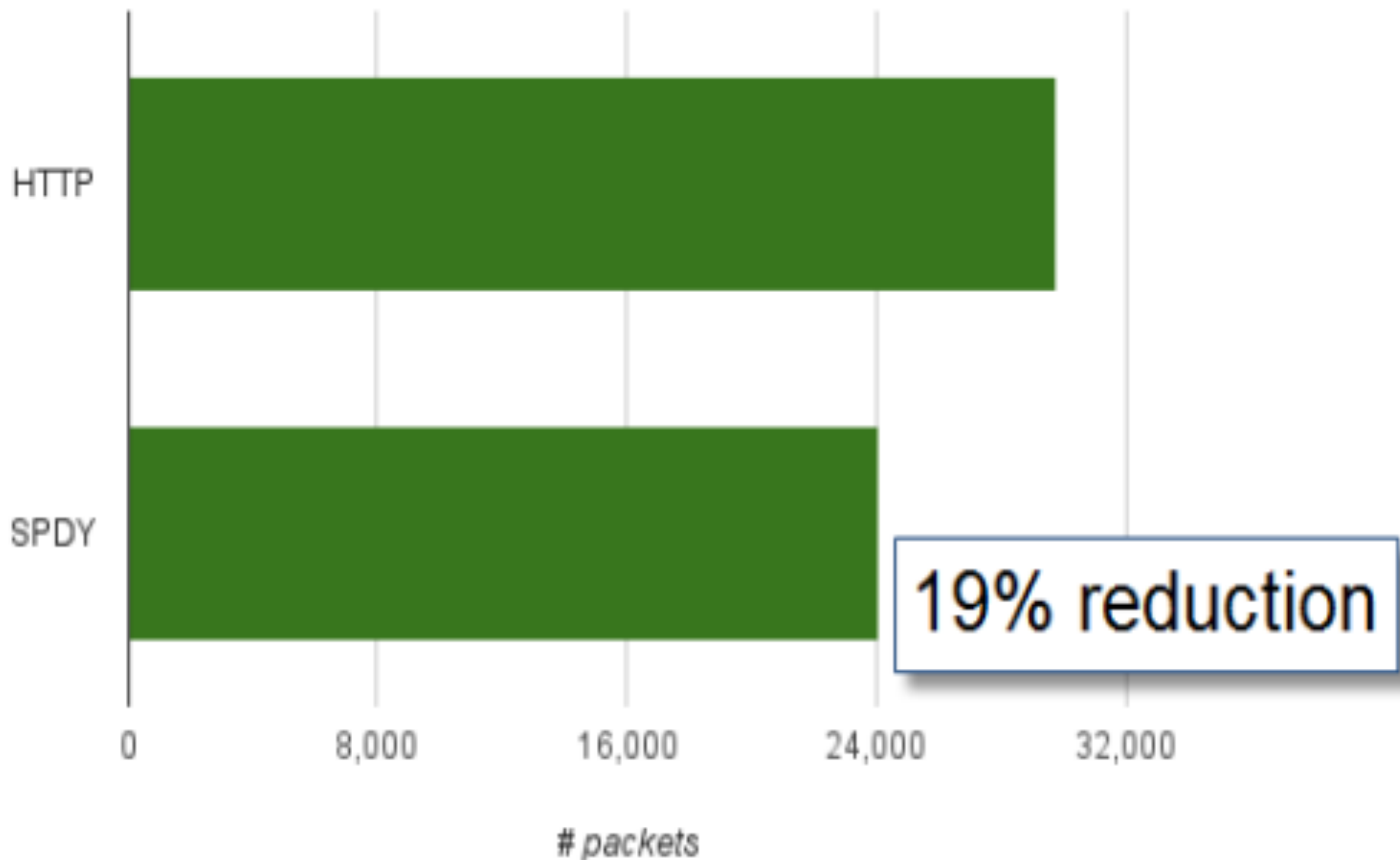
SPDY "Less is More"
Uplink data



SPDY "Less is More" Downlink data

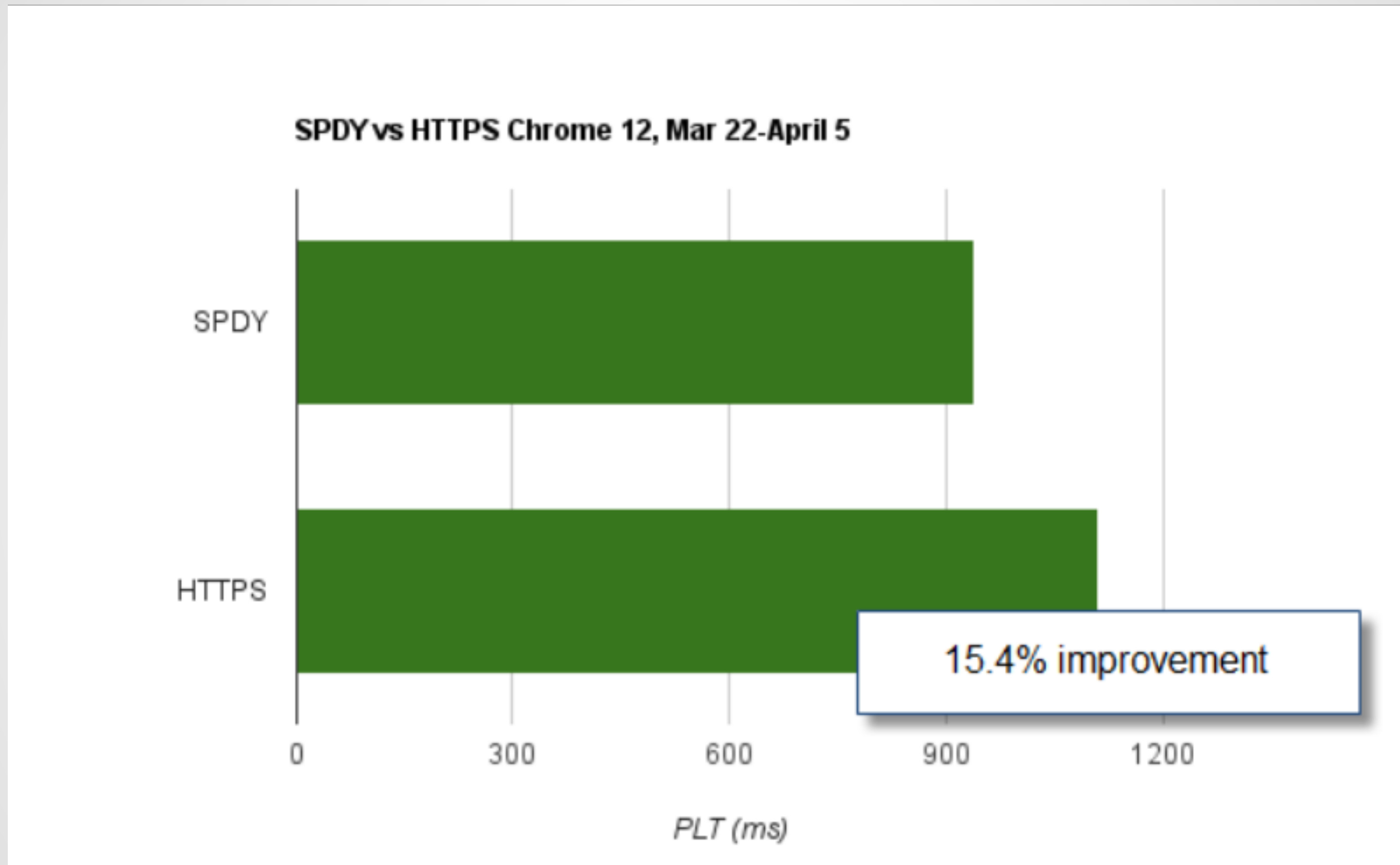


SPDY "Less is More"
Total Packets



Performance Results

Google Results



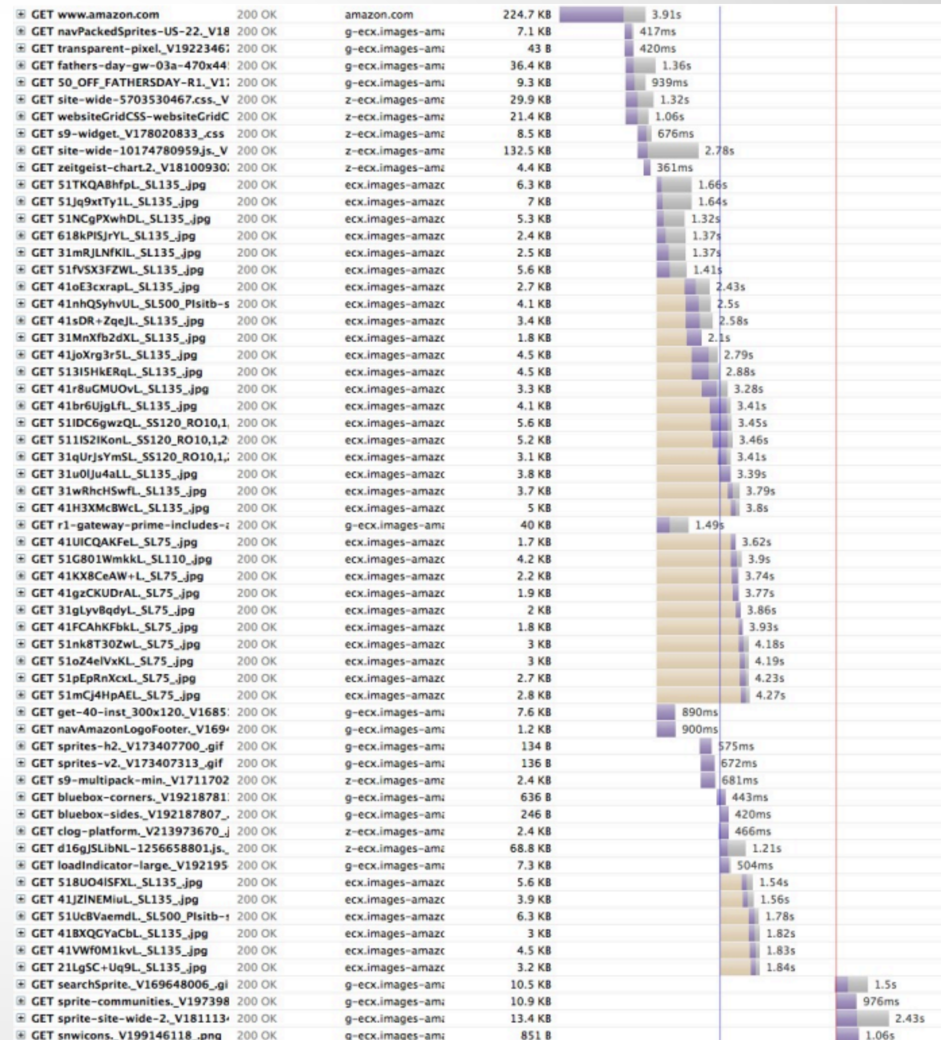
Cotendo Tests

Amazon.com home

HTTP

3G AT&T
~200ms RTT

PLT: **12.50 secs**



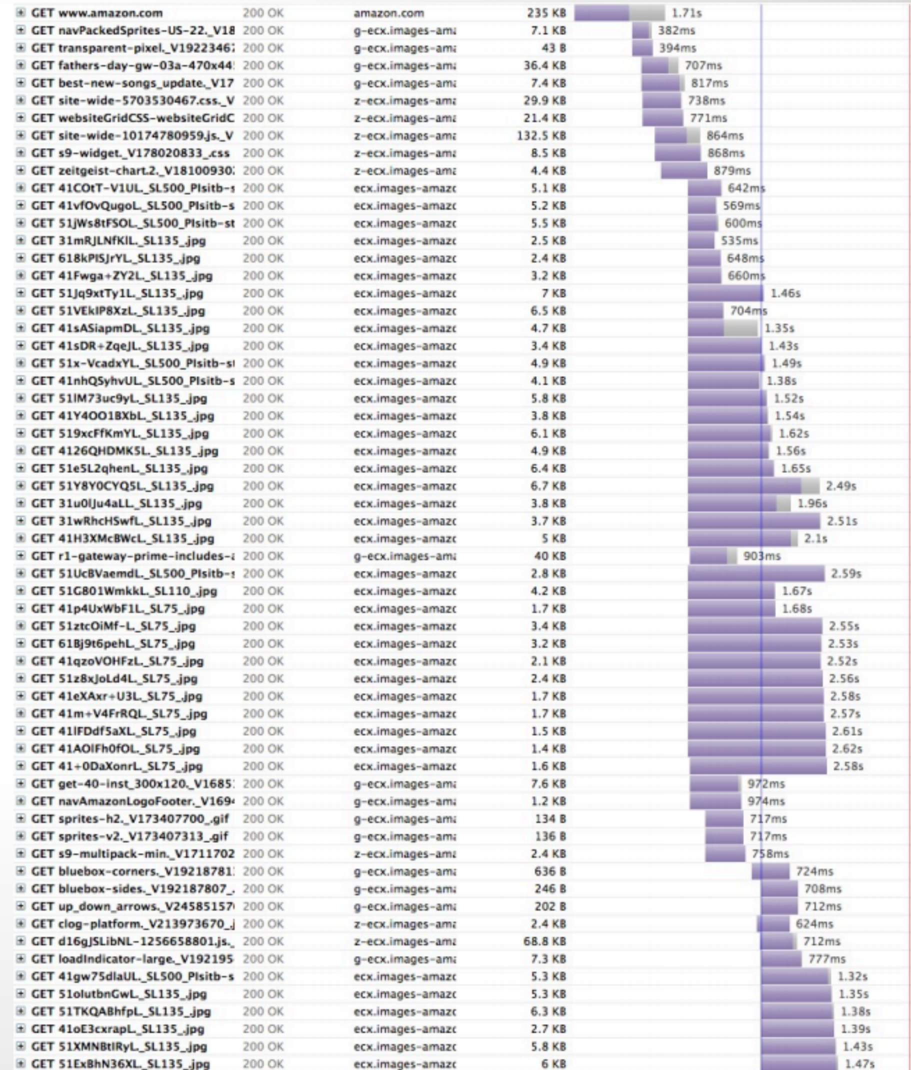
Cotendo Tests

Amazon.com home

SPDY

3G AT&T
~200ms RTT

PLT: **6.26 secs**
-49%



GET www.amazon.com	200 OK	amazon.com	235 KB	1.71s
GET navPackedSprites-US-22._V18	200 OK	g-ecx.images-am	7.1 KB	382ms
GET transparent-pixel._V1922346	200 OK	g-ecx.images-am	43 B	394ms
GET fathers-day-gw-03a-470x44	200 OK	g-ecx.images-am	36.4 KB	707ms
GET best-new-songs_update._V17	200 OK	g-ecx.images-am	7.4 KB	817ms
GET site-wide-5703530467.css._V	200 OK	z-ecx.images-am	29.9 KB	738ms
GET websiteGridCSS-websiteGridC	200 OK	z-ecx.images-am	21.4 KB	771ms
GET site-wide-10174780959.js._V	200 OK	z-ecx.images-am	132.5 KB	864ms
GET s9-widget._V178020833_.css	200 OK	z-ecx.images-am	8.5 KB	868ms
GET zeitgeist-chart.2._V18100930	200 OK	z-ecx.images-am	4.4 KB	879ms
GET 41COtT-V1UL._SL500_PIsitb-	200 OK	ecx.images-amaz	5.1 KB	642ms
GET 41vfOvQugoL._SL500_PIsitb-s	200 OK	ecx.images-amaz	5.2 KB	569ms
GET 51jWs8tFSOL._SL500_PIsitb-st	200 OK	ecx.images-amaz	5.5 KB	600ms
GET 31mRJLNfKlL._SL135_.jpg	200 OK	ecx.images-amaz	2.5 KB	535ms
GET 618kPISJrYL._SL135_.jpg	200 OK	ecx.images-amaz	2.4 KB	648ms
GET 41Fwga+ZY2L._SL135_.jpg	200 OK	ecx.images-amaz	3.2 KB	660ms
GET 51Jq9xtTy1L._SL135_.jpg	200 OK	ecx.images-amaz	7 KB	1.46s
GET 51VEkIP8XzL._SL135_.jpg	200 OK	ecx.images-amaz	6.5 KB	704ms
GET 41sASiapmDL._SL135_.jpg	200 OK	ecx.images-amaz	4.7 KB	1.35s
GET 41sDR+ZqeJL._SL135_.jpg	200 OK	ecx.images-amaz	3.4 KB	1.43s
GET 51x-VcadxYL._SL500_PIsitb-s	200 OK	ecx.images-amaz	4.9 KB	1.49s
GET 41nhQSyhvUL._SL500_PIsitb-s	200 OK	ecx.images-amaz	4.1 KB	1.38s
GET 51lM73uc9yL._SL135_.jpg	200 OK	ecx.images-amaz	5.8 KB	1.52s
GET 41Y4OO1BXbL._SL135_.jpg	200 OK	ecx.images-amaz	3.8 KB	1.54s
GET 519xcFfKmYL._SL135_.jpg	200 OK	ecx.images-amaz	6.1 KB	1.62s
GET 4126QHDMK5L._SL135_.jpg	200 OK	ecx.images-amaz	4.9 KB	1.56s
GET 51e5L2qhenL._SL135_.jpg	200 OK	ecx.images-amaz	6.4 KB	1.65s
GET 51Y8Y0CYQ5L._SL135_.jpg	200 OK	ecx.images-amaz	6.7 KB	2.49s
GET 31u0lJu4aLL._SL135_.jpg	200 OK	ecx.images-amaz	3.8 KB	1.96s
GET 31wRhcHSwfL._SL135_.jpg	200 OK	ecx.images-amaz	3.7 KB	2.51s
GET 41H3XMcBWcL._SL135_.jpg	200 OK	ecx.images-amaz	5 KB	2.1s
GET r1-gateway-prime-includes-	200 OK	g-ecx.images-am	40 KB	903ms
GET 51UcBVaemdL._SL500_PIsitb-	200 OK	ecx.images-amaz	2.8 KB	2.59s
GET 51G801WmkkL._SL110_.jpg	200 OK	ecx.images-amaz	4.2 KB	1.67s
GET 41p4UxWbF1L._SL75_.jpg	200 OK	ecx.images-amaz	1.7 KB	1.68s
GET 51ztcOiMf-L._SL75_.jpg	200 OK	ecx.images-amaz	3.4 KB	2.55s
GET 61Bj9t6pehL._SL75_.jpg	200 OK	ecx.images-amaz	3.2 KB	2.53s
GET 41qzoVOHFzL._SL75_.jpg	200 OK	ecx.images-amaz	2.1 KB	2.52s
GET 51z8xJoLd4L._SL75_.jpg	200 OK	ecx.images-amaz	2.4 KB	2.56s
GET 41eXAxr+U3L._SL75_.jpg	200 OK	ecx.images-amaz	1.7 KB	2.58s
GET 41m+V4FrRQL._SL75_.jpg	200 OK	ecx.images-amaz	1.7 KB	2.57s
GET 41lFDdf5aXL._SL75_.jpg	200 OK	ecx.images-amaz	1.5 KB	2.61s
GET 41AOIFh0fOL._SL75_.jpg	200 OK	ecx.images-amaz	1.4 KB	2.62s
GET 41+0DaXonrL._SL75_.jpg	200 OK	ecx.images-amaz	1.6 KB	2.58s
GET get-40-inst_300x120._V1685	200 OK	g-ecx.images-am	7.6 KB	972ms
GET navAmazonLogoFooter._V169	200 OK	g-ecx.images-am	1.2 KB	974ms
GET sprites-h2._V173407700_.gif	200 OK	g-ecx.images-am	134 B	717ms
GET sprites-v2._V173407313_.gif	200 OK	g-ecx.images-am	136 B	717ms
GET s9-multipack-min._V1711702	200 OK	z-ecx.images-am	2.4 KB	758ms
GET bluebox-corners._V19218781	200 OK	g-ecx.images-am	636 B	724ms
GET bluebox-sides._V192187807_.	200 OK	g-ecx.images-am	246 B	708ms
GET up_down_arrows._V24585157	200 OK	g-ecx.images-am	202 B	712ms
GET clog-platform._V213973670_.	200 OK	z-ecx.images-am	2.4 KB	624ms
GET d16gJSLibNL-1256658801.js._	200 OK	z-ecx.images-am	68.8 KB	712ms
GET loadIndicator-large._V192195	200 OK	g-ecx.images-am	7.3 KB	777ms
GET 41gw75dlaUL._SL500_PIsitb-s	200 OK	ecx.images-amaz	5.3 KB	1.32s
GET 51olutbnGwL._SL135_.jpg	200 OK	ecx.images-amaz	5.3 KB	1.35s
GET 51TKQABhfpL._SL135_.jpg	200 OK	ecx.images-amaz	6.3 KB	1.38s
GET 41oE3cxrapL._SL135_.jpg	200 OK	ecx.images-amaz	2.7 KB	1.39s
GET 51XMNBtIRyL._SL135_.jpg	200 OK	ecx.images-amaz	5.8 KB	1.43s
GET 51ExBhN36XL._SL135_.jpg	200 OK	ecx.images-amaz	6 KB	1.47s

Other results

- Firefox confirmed Chrome results
- Google recently reported that SPDY over SSL is now faster than HTTP without SSL
- BoostEdge paper confirms Google numbers
- *need vendors to publish more!*

Deployment
A Process of Elimination

- Transport choices: TCP or UDP
  - Chose TCP
- Port choices: 80 or 443
  - But both are taken!
- Chrome test shows usability of port 80 for non HTTP protocols is <75%.
  - Using port 80 makes SPDY like Pipelining.
- Port 443 is the only untampered port.
- Other ports: blocked by firewalls

Pause - That was the Big Picture
"Better is the enemy of good"

- The aforementioned items are the non-controversial parts of SPDY.
- HTTP/2.0 should take those concepts.
- Minutiae doesn't matter:
  - exact framing syntax
  - exact compression algorithm
- Stay Focused on the Big Picture!

Why Nots?

Why not SCTP?

Multiplexing over a single TCP stream does have one element of head-of-line blocking.

But SCTP has problems:

- Not available on most platforms
- Requires administrative privs to install (so it can't be bundled easily with browser installs)
- Incompatible with NAT on today's internet.

Why not Pipelining?

Pipelining was introduced a decade ago.

- Wasn't deployable due to intermediaries that didn't handle it properly.
- It has complex head-of-line blocking problems (hanging GETs)
- Firefox team list of heuristics is huge. SPDY was easier to build than pipelining.
- Counterpoint: mobile uses pipelining. Does it work?

Why 1 Connection?

- More efficient for network, memory use, and server scalability; better compression.
- Don't have to wait for a handshake to complete before sending a request.
- Doesn't encourage Buffer bloat. (Jim Gettys)
- Lets the transport do what it does best.
- Would like to see more research here.

SPDY for Mobile

Mobile is Different

- New client-side problems
  - Battery life constraints
  - Small CPUs (changing fast!)
- New Network Properties
  - Latency from 150 - 300ms per Round Trip
  - Bandwidth 1-4Mbps
- New use cases
  - Mobile Web Browsers are 1st generation
    - So web browsing sucks
  - Everyone uses Apps w/ REST APIs anyway

SPDY and Mobile

- Fewer connections/bytes/packets reduces transmit requirements of radio
- Mobile connection management is different due to NAT and in-and-out networks.
    - Can't use TCP Keepalives
    - PING frame detects closed conns quickly
- Header compression minimizes upstream sends
- 1 conn per domain minimizes tcp-level control traffic

The Tough Stuff

Don't make things "optional"

- Optional features are disabled features.
  e.g. pipelining.
- Optional features are buggy.
  e.g. absolute URIs fail on many HTTP/1.1 servers.
- Feature detection often takes a round-trip.
  e.g. does it support a compressed request?
- Proxies will tamper with option negotiation.
  e.g. Accept-EnXcoding

Security

I often hear that security is difficult/expensive/costly or unwanted.

I've NEVER heard this complaint from a user.

I've ONLY heard this complaint from proxy and server *implementors*.

Could it be that users just expect it to be secure?

What Security Can HTTP/2.0 Provide?

- Security is accomplished across the stack, not at a single layer. But HTTP does play a role.

- Requiring SSL with HTTP/2.0 will:
  - Protect the user from eavesdroppers (firesheep!)
  - Protect from content tampering
  - Protect the protocol for future extensions
  - Authenticate servers

Insecure Protocols Hurt Users

Without integrity & privacy, you enable anyone to:

- record data about you
- inject advertisements into your content
- prevent access to certain sites
- alter site content
- limit your bandwidth (for any reason)

Is this what the user wants?

Insecure Protocols Enable Transparent Proxies

- Transparent proxies are proxies that you didn't opt-in to
  - As a site operator, they can alter your content
  - As a user, they can alter your web experience
- Transparent proxies are to blame for many of our protocol woes:
  - Inability to fix HTTP/1.1 pipelining
  - Turning off compression behind the user's back
- They are easy to deploy, however...

SSL is not Expensive

- Twitter and Google rolled out with zero additional hardware.
- Bulk encryption (RC4) is basically free
- Handshakes are a little expensive, but <1% of CPU costs
- Certificates are free.
- SPDY + SSL is faster than HTTP.

Is an insecure protocol legal anymore?

- Privacy laws in the US & EU make those that leak private information liable for the losses

- Should web site administrators need to know how HTTP works in order to obey basic laws?

Recognizing Different Requirements

We need multiple protocols, not options

We have distinct use cases

- End User HTTP
  - targets consumers and Internet User needs
- BackOffice HTTP
  - for those using HTTP in behind their own firewalls
- Caching HTTP (also corp firewall HTTP)
  - For corporate environments or organizations sharing a common cache
  - May not be a separate protocol, but lets make it work explicitly.

End User HTTP

- Optimized for the Internet Consumer.
- Features:
  - Always secure (safe to use in the Cafe)
  - Always compressed
  - Always fast

BackOffice HTTP

- Used for backoffice server infrastructure, already behind your own firewalls.
- Features:
  - Not implemented by browsers
  - Makes SSL optional
  - Makes Compression optional

Caching HTTP

- Used by corporations with filtering firewalls or those that want to have an external cache
- Features:
  - User opts-in. Never transparent.
  - SSL to the proxy; proxy brokers the request to origin
  - Respects HSTS
  - Reduces need for SSL MITM

Thank you!

Looking forward to a fantastic HTTP/2.0!