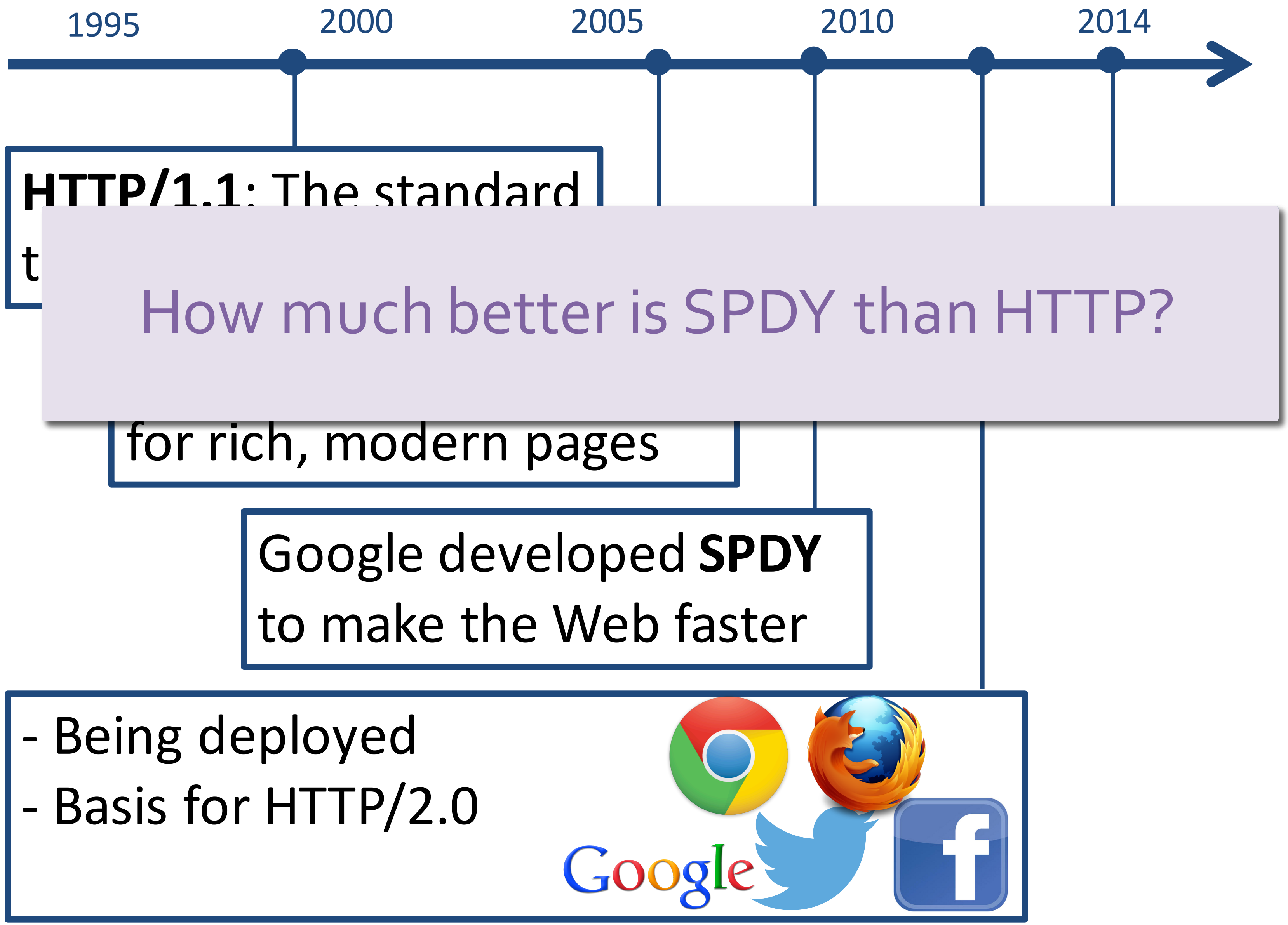
How speedy is SPDY?

Xiao (Sophia) Wang

University of Washington

Facebook Inc.



HTTP/1.1: The standard

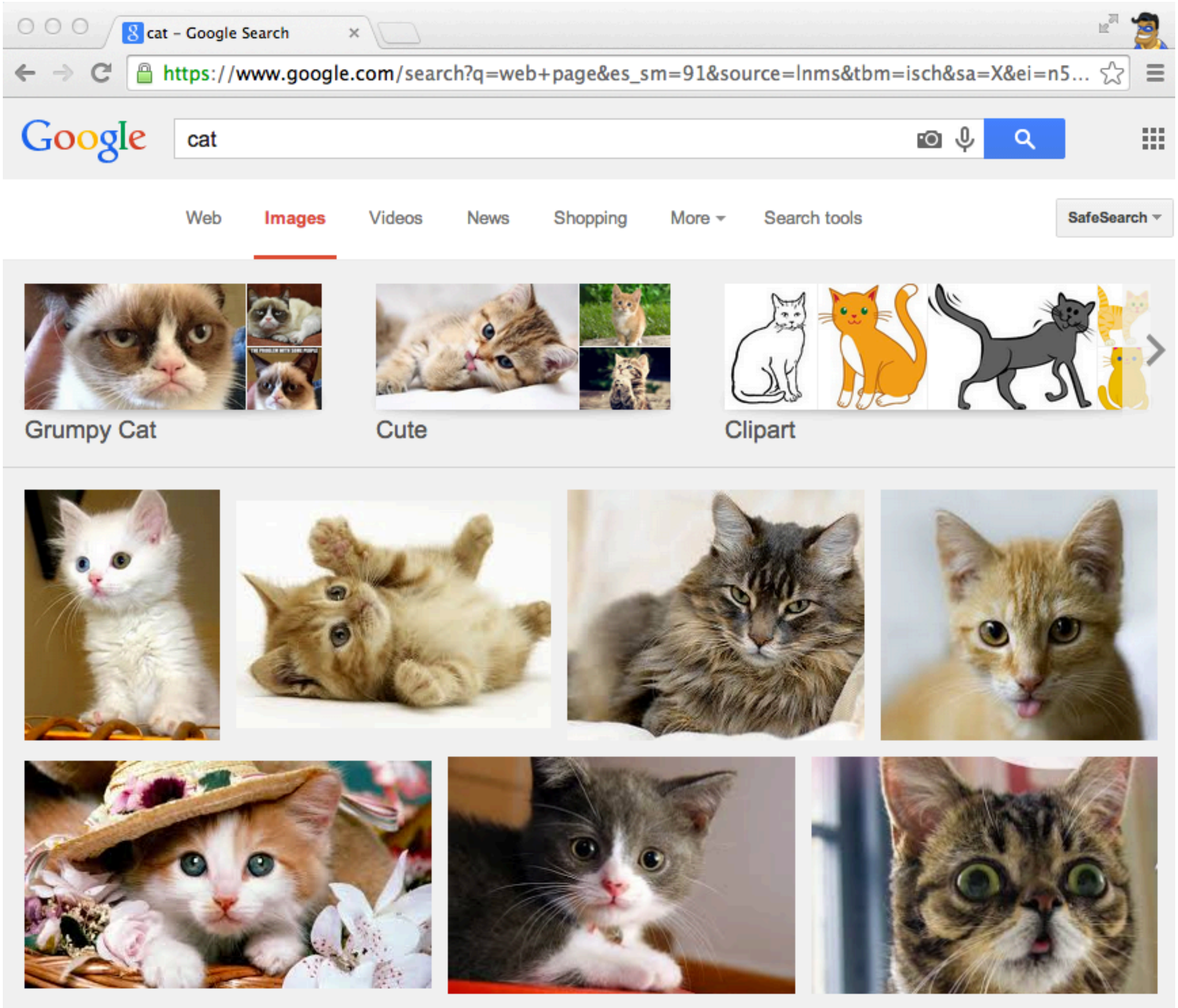
t

How much better is SPDY than HTTP?

for rich, modern pages

Google developed **SPDY** to make the Web faster

- Being deployed
- Basis for HTTP/2.0



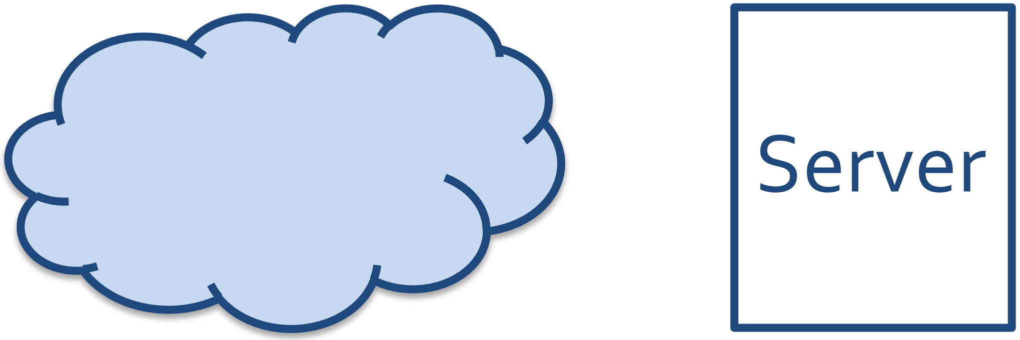
cat - Google Search

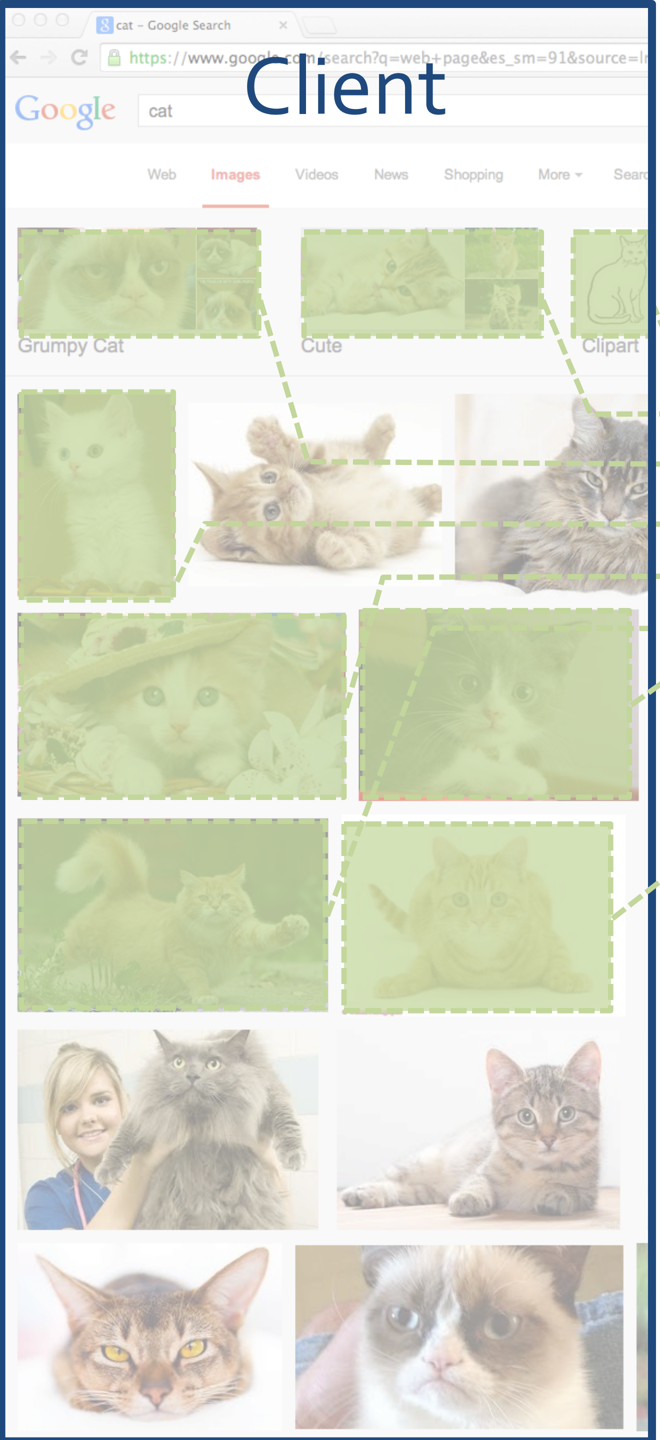
https://www.google.com/search?q=web+page&es_sm=91&source=lnms&tbm=isch&sa=X&ei=n5...

cat

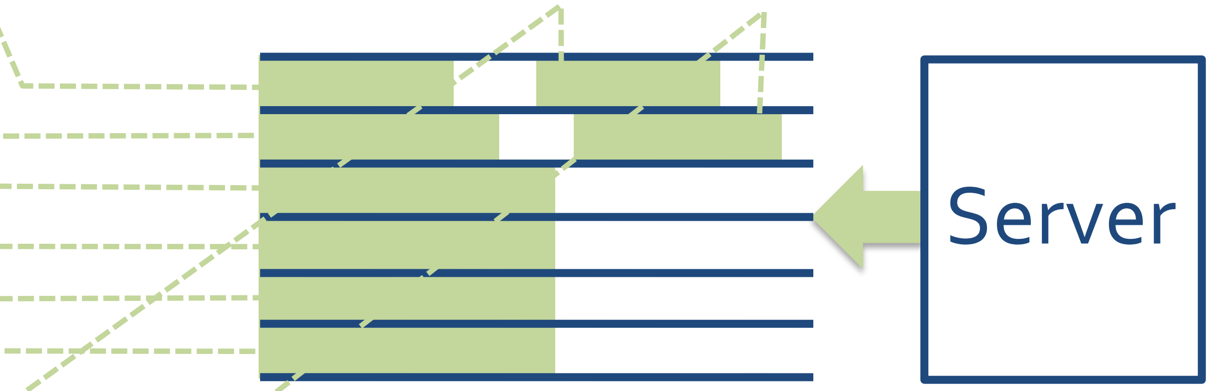
HTTP/1.1 problems

Client



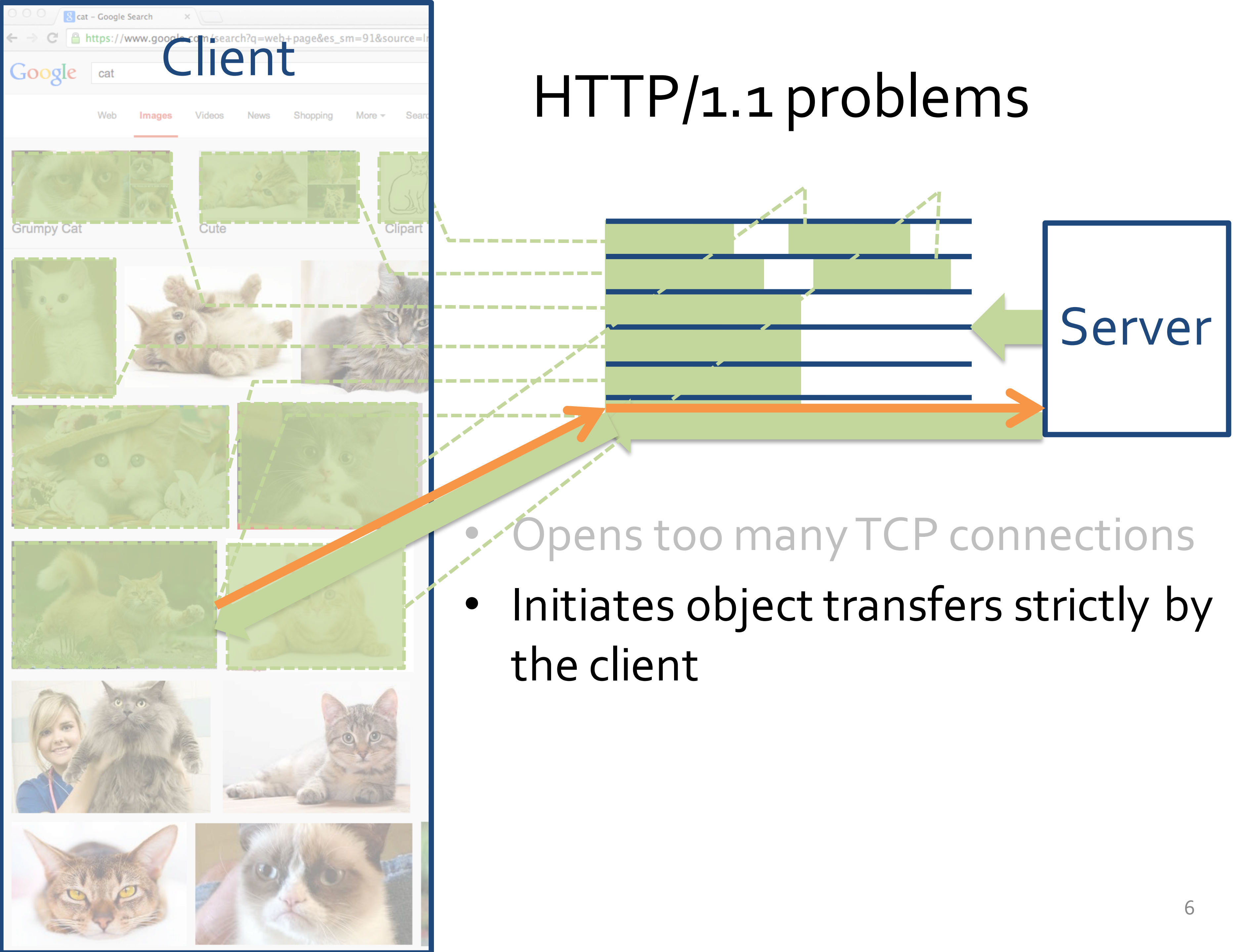


HTTP/1.1 problems



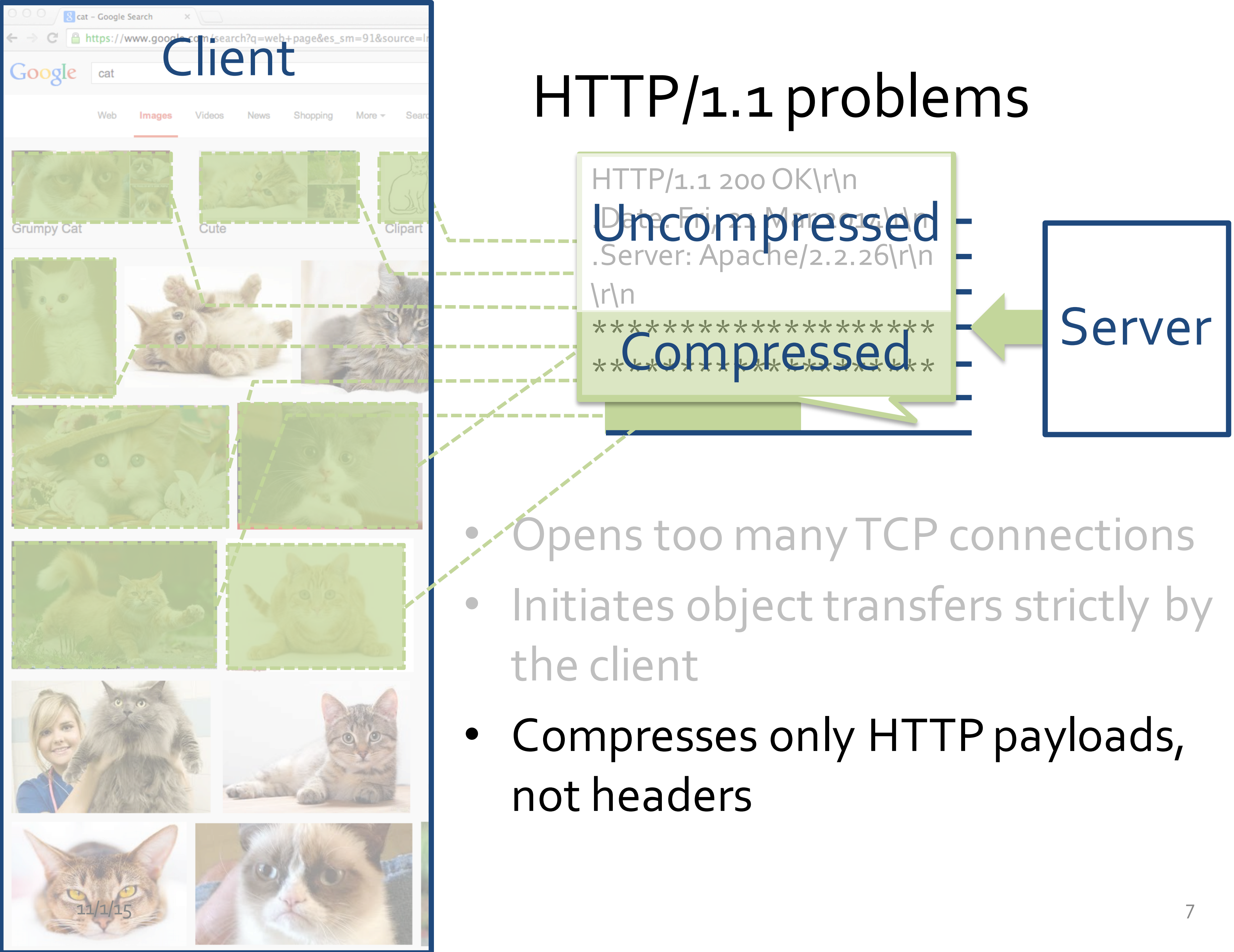
- Opens too many TCP connections

HTTP/1.1 problems



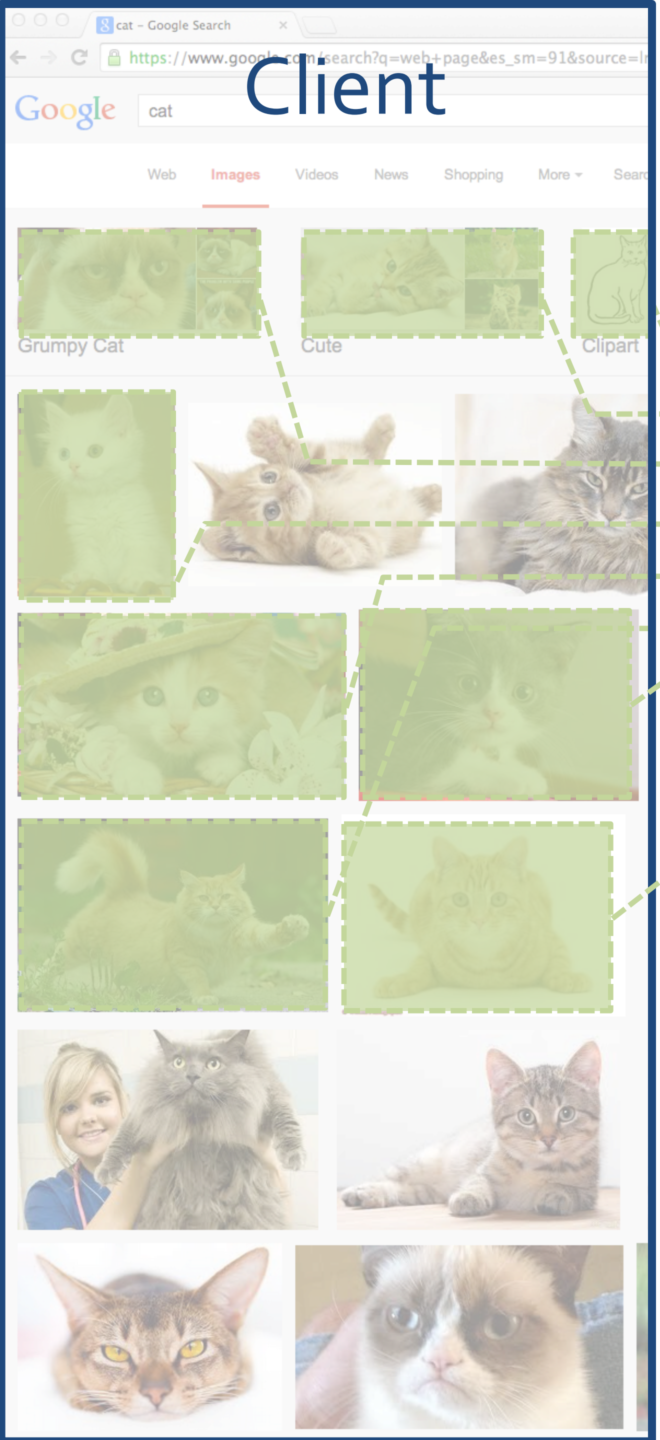
- Opens too many TCP connections
- Initiates object transfers strictly by the client

HTTP/1.1 problems



HTTP/1.1 200 OK\r\n
.Date: Fri, 21 Mar 2014\r\n
.Server: Apache/2.2.26\r\n
\r\n
****
****

- Opens too many TCP connections
- Initiates object transfers strictly by the client
- Compresses only HTTP payloads, not headers



Client

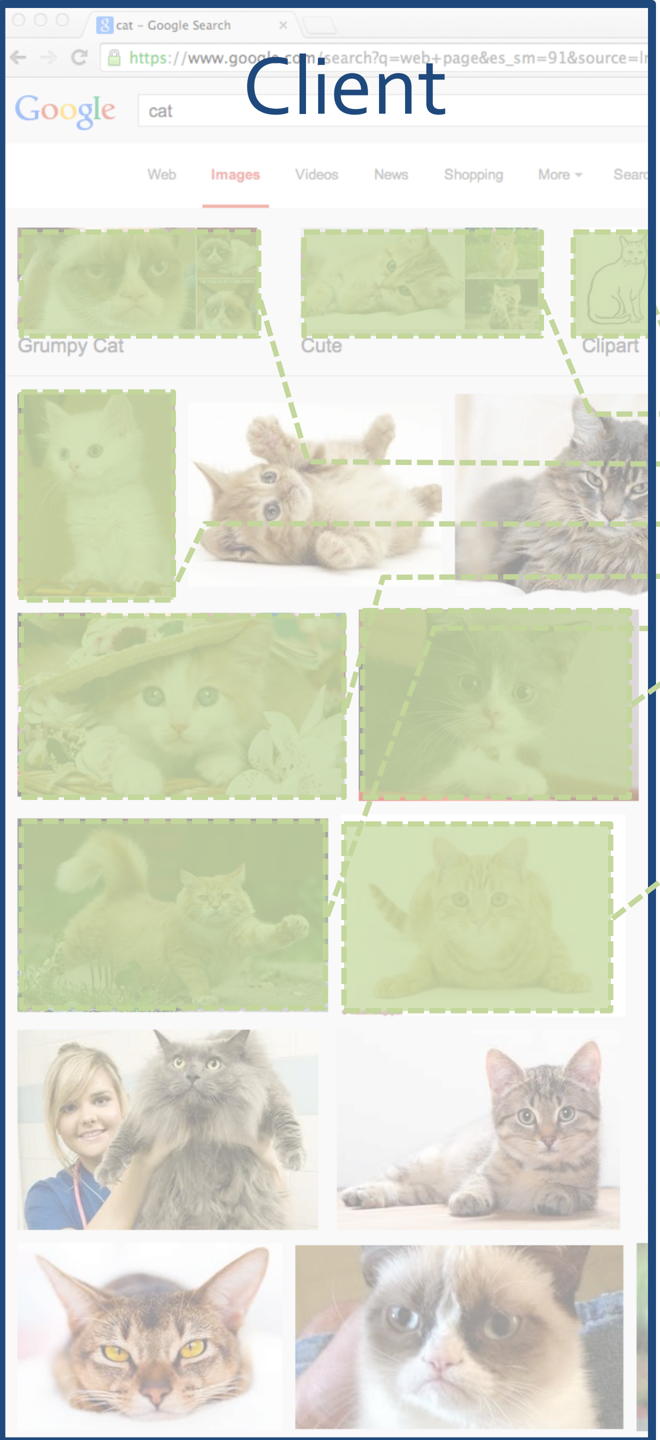
HTTP/1.1 problems

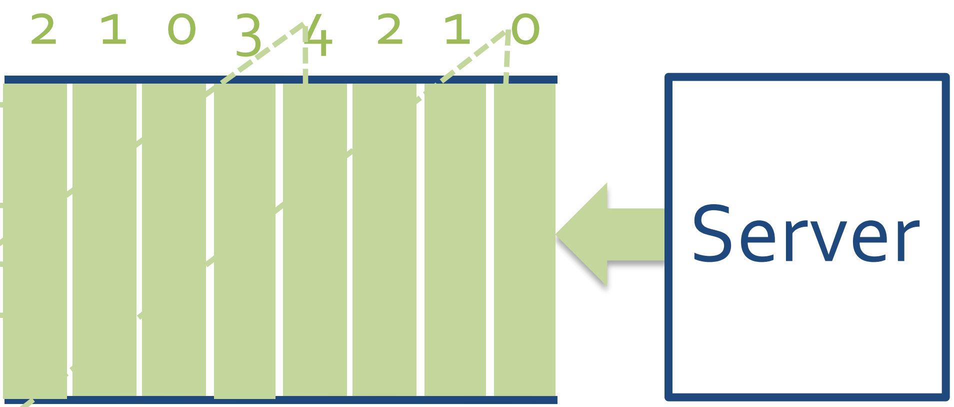
HTTP/1.1 200 OK\r\n

SPDY is proposed to address these issues

- Opens too many TCP connections
- Initiates object transfers strictly by the client
- Compresses only HTTP payloads, not headers

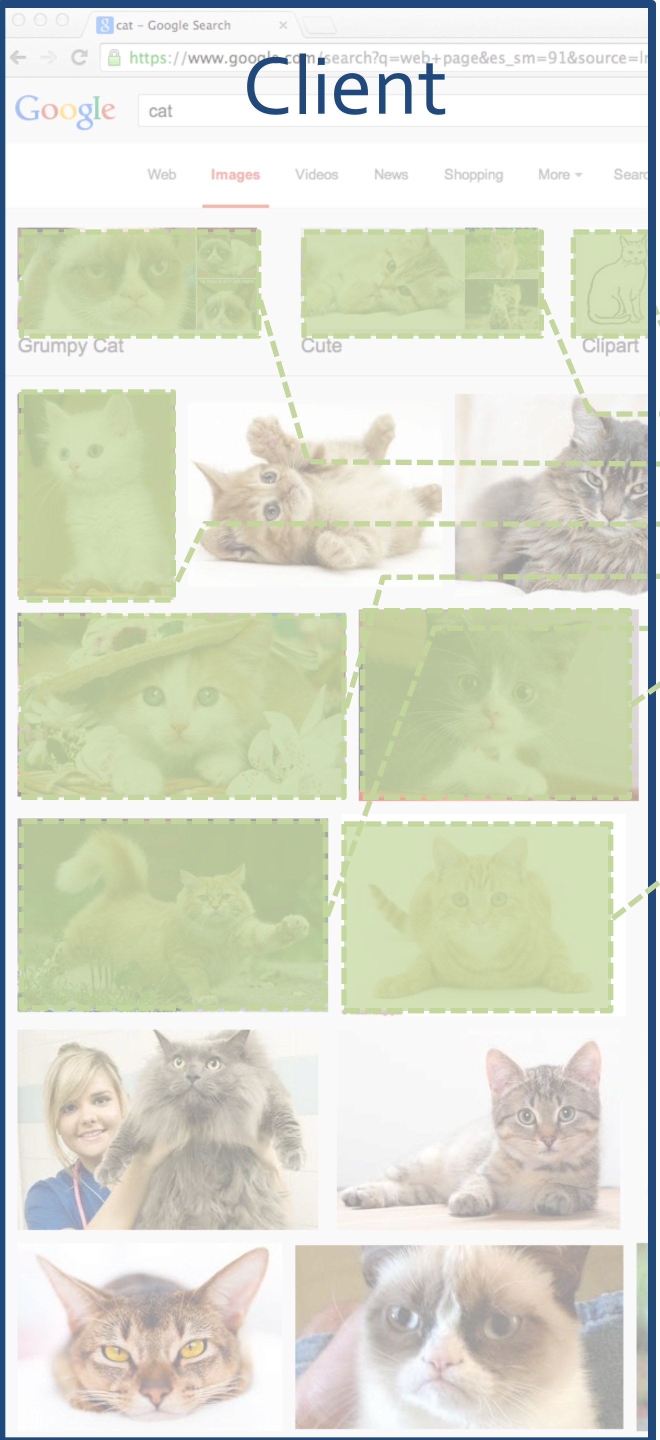
SPDY

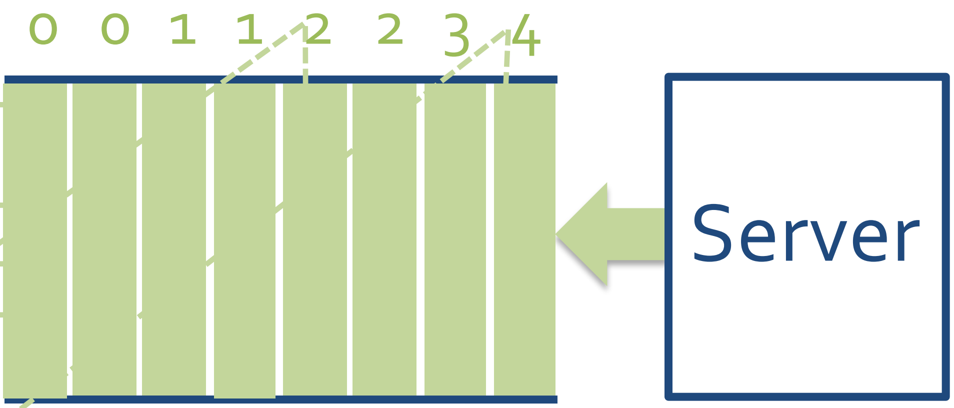




- ~~Opens too many TCP connections~~
- Multiplexes sliced frames into a single TCP connection

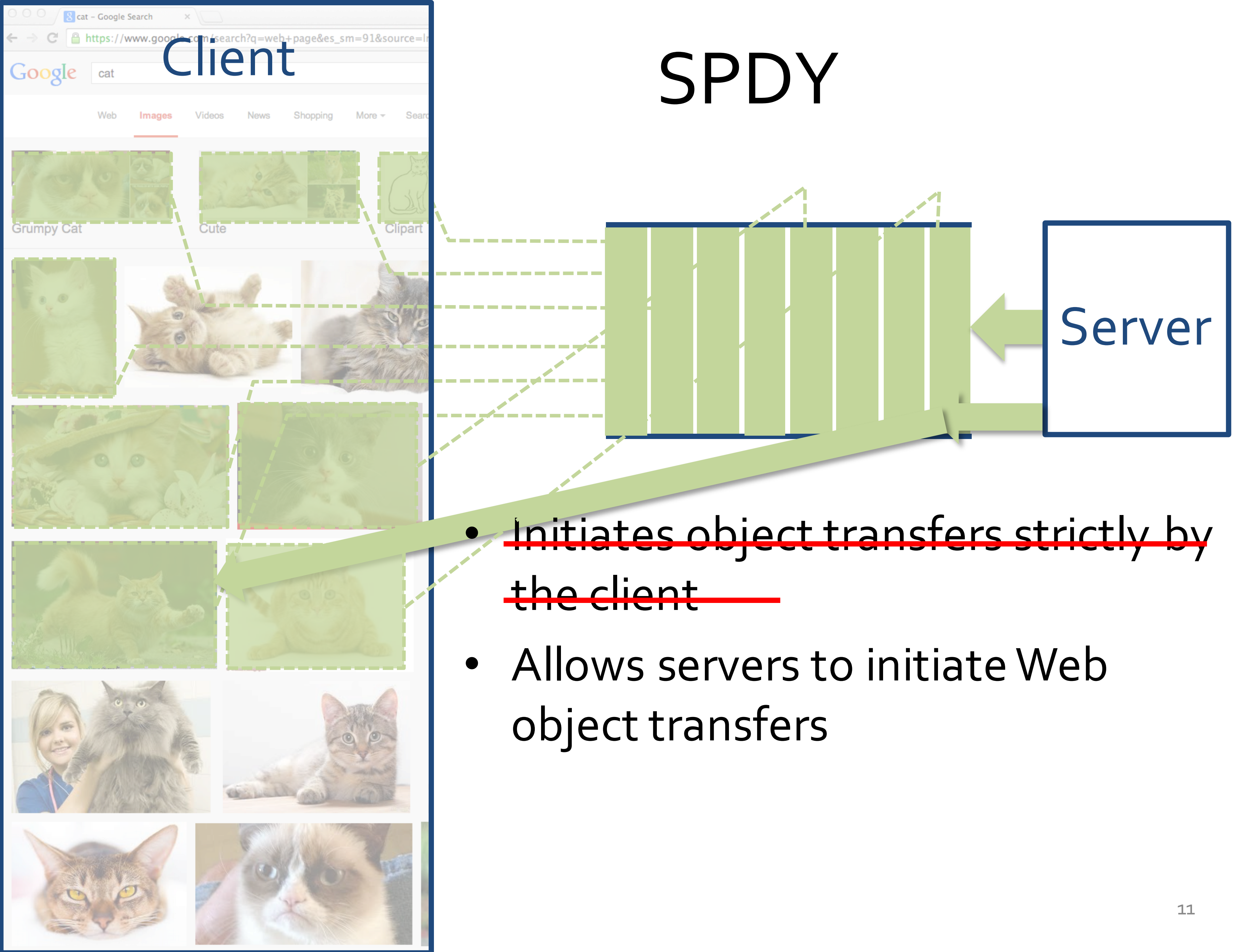
SPDY





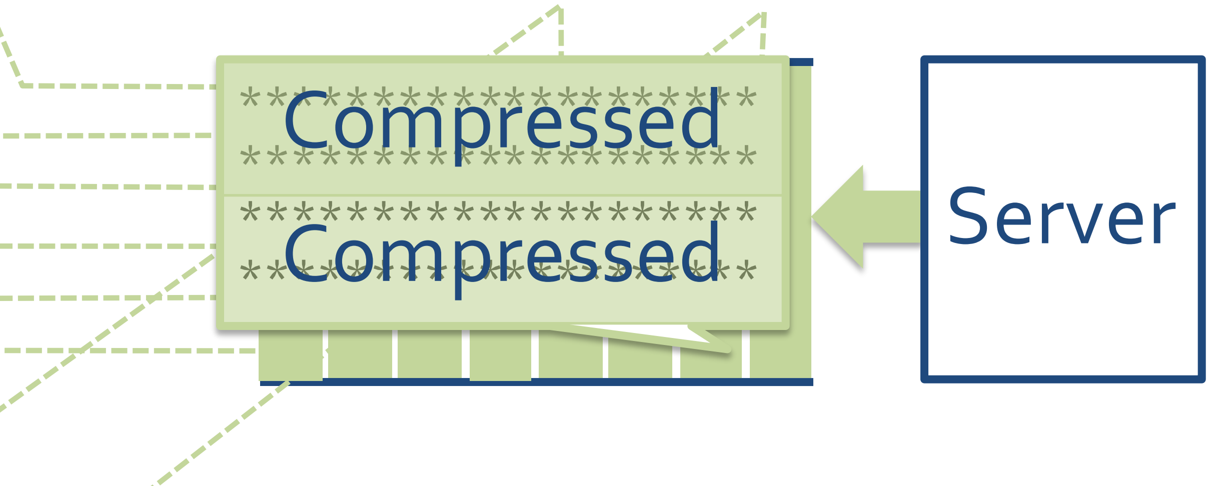
- ~~Opens too many TCP connections~~
- Multiplexes sliced frames into a single TCP connection
- Prioritizes Web objects

SPDY



- ~~Initiates object transfers strictly by the client~~
- Allows servers to initiate Web object transfers

SPDY



- ~~Compresses only HTTP payloads, not headers~~
- Compresses both HTTP payloads and headers

How well does SPDY perform?

Google

SPDY helps **27%** to **60%**

How well does SPDY perform?

Google

SPDY helps **27%** to **60%**

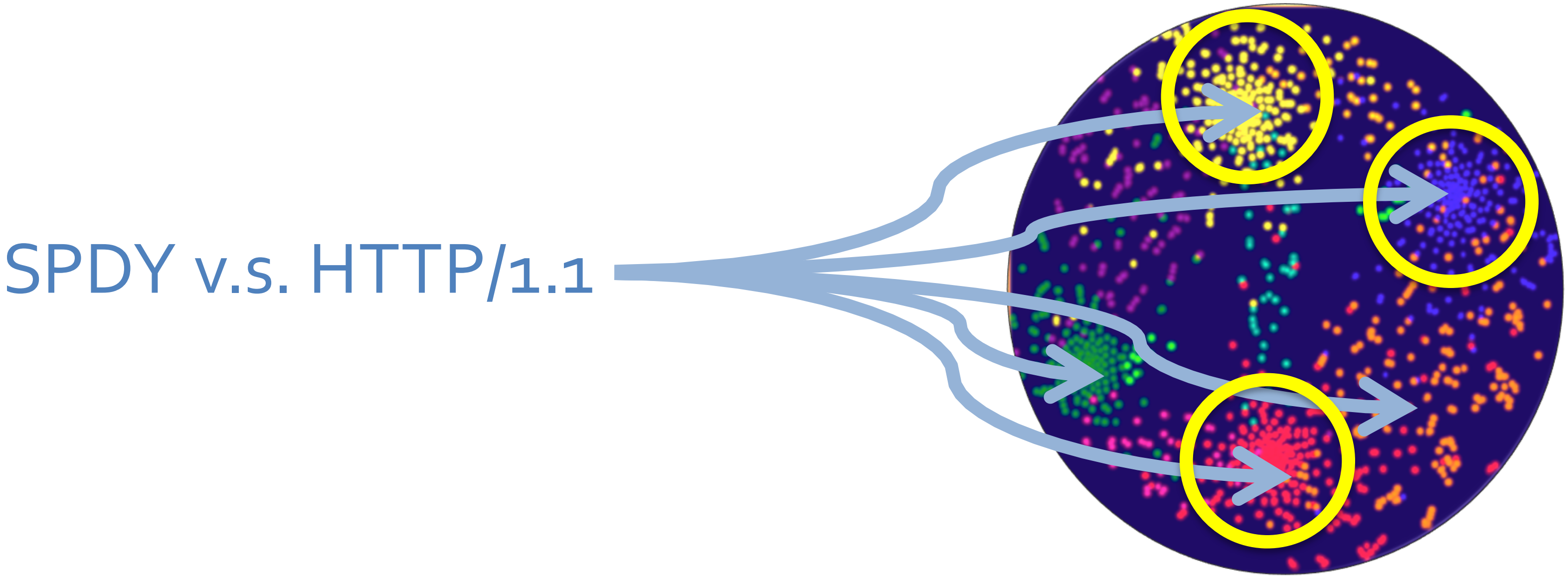
Microsoft

SPDY sometimes helps and sometimes hurts.

Measurement results conflict

Goals

- A systematic study of SPDY that
  - Extensively sweeps the parameter space
  - Links SPDY performance to underlying factors
  - Identifies the dominant factors

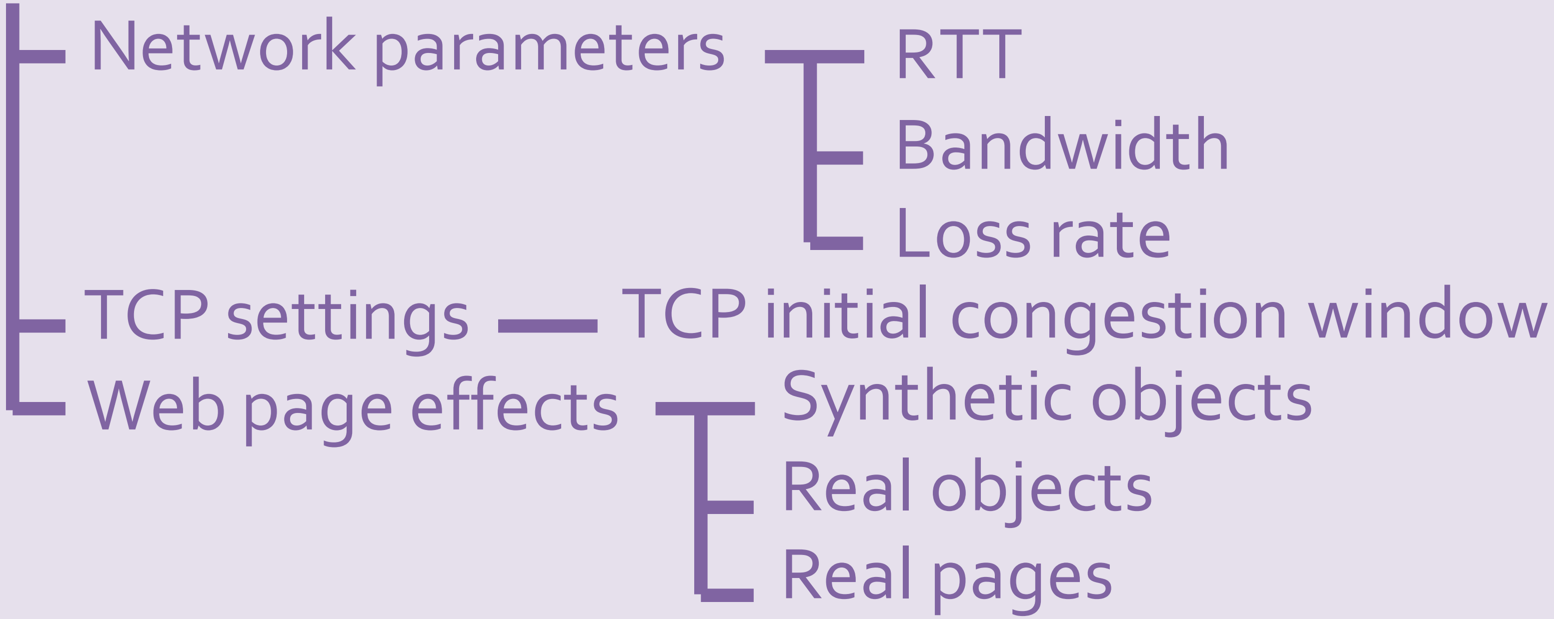


Challenge

Many factors **external** to SPDY affect SPDY

Approach

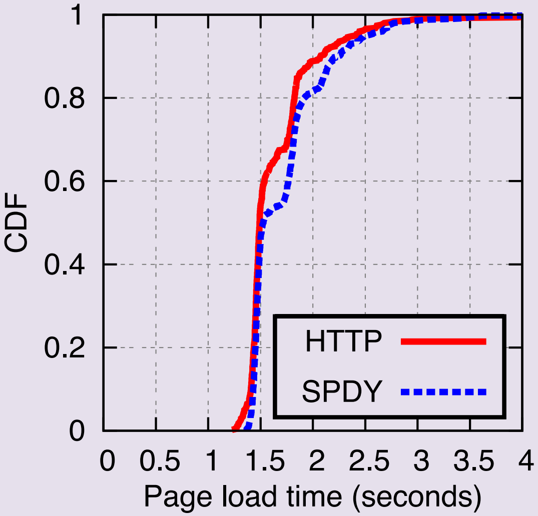
Isolate factors, sweep the parameter space



- Network parameters
  - RTT
  - Bandwidth
  - Loss rate
- TCP settings
  - TCP initial congestion window
- Web page effects
  - Synthetic objects
  - Real objects
  - Real pages

Challenge

Page load time has high variance



Variance: **0.5** second

Difference: **0.02** second

Approach

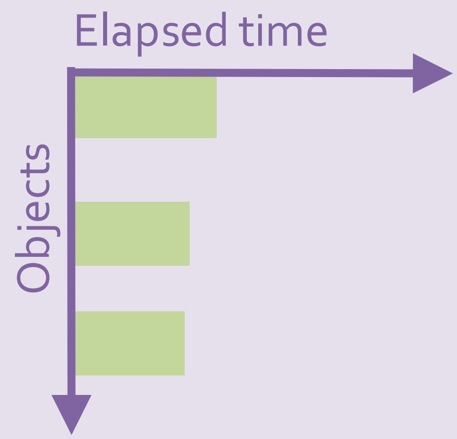
Control source of variability by

- Experimenting in a **controlled network**
- Using **our emulator** instead of browsers

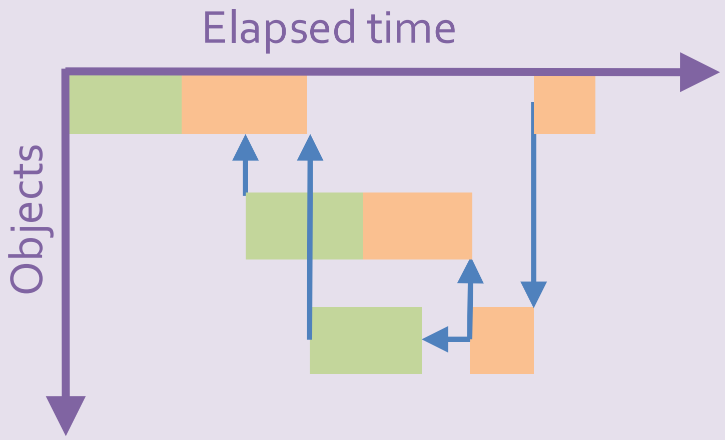
Challenge

Dependencies between network and browser computation affect page loads

No browser



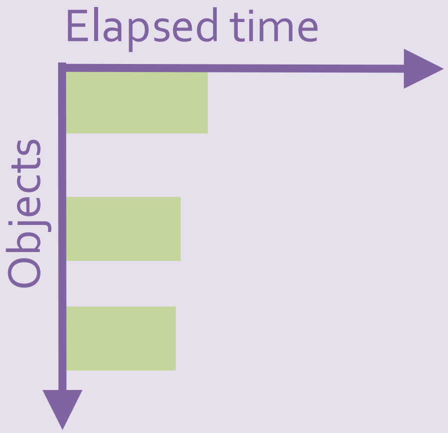
Browser computation



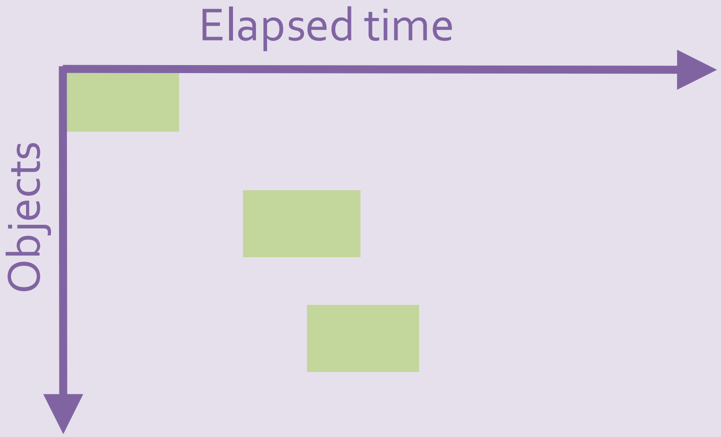
Challenge

Dependencies between network and browser computation affect page loads

No browser



Browser computation



Approach

Preserve dependencies.

Outline

- Understanding SPDY's performance with
  - Synthetic objects
  - Real objects
  - Real pages
- Enhanced policies for SPDY

Outline

- Understanding SPDY's performance with
  - **Synthetic objects**
  - Real objects
  - Real pages
- Enhanced policies for SPDY

Extensively sweep parameter space

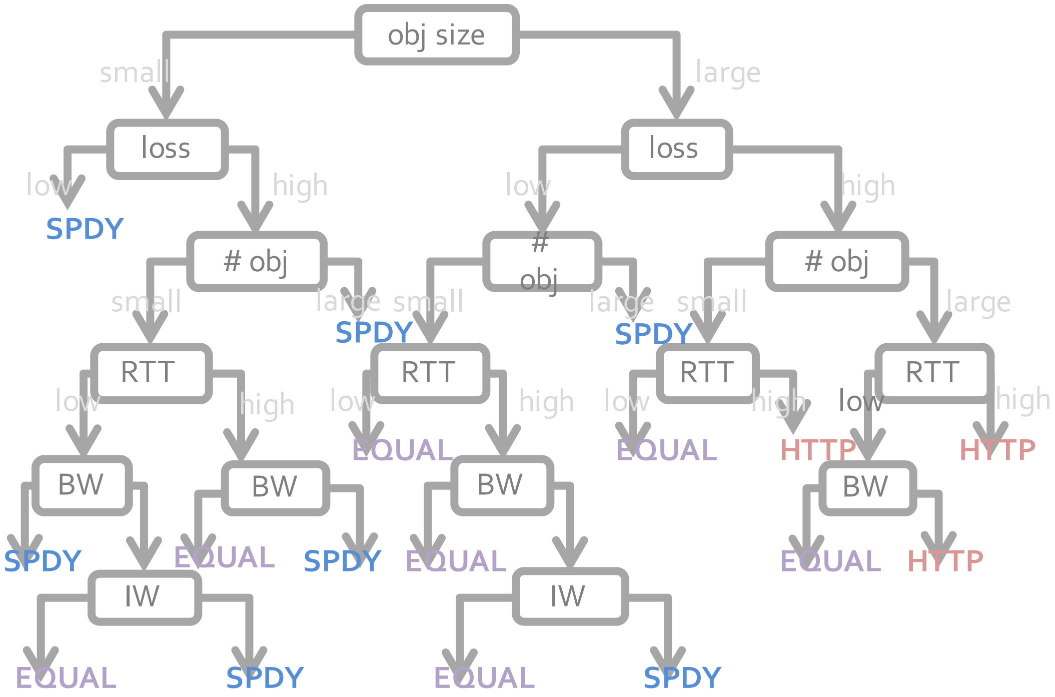
	Factors	Range
Network parameters	RTT	20ms, 100ms, 200ms
	Bandwidth	1Mbps, 10Mbps
	Loss rate	0, .5%, 1%, 2%
TCP settings	TCP IW	3, 10, 21, 32
Synthetic objects	Web obj. size	100B, 1K, 10K, 100K, 1M
	# of objects	2, 8, 16, 32, 64, 128, 512

Make HTTP requests

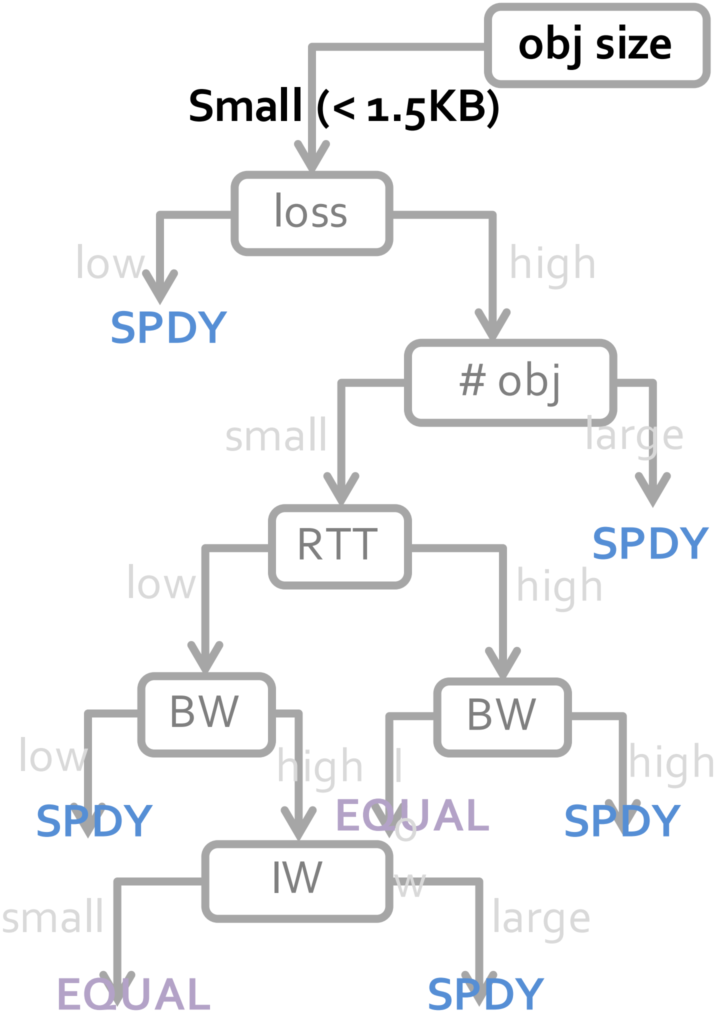
Link SPDY performance to factors

→ Decision tree analysis

Six factors, Thousands of data points



SPDY helps on small objects



obj size

Small (< 1.5KB)

loss

low

SPDY

high

# obj

small

large

SPDY

RTT

low

high

BW

low

SPDY

high

IW

small

EQUAL

large

SPDY

BW

low

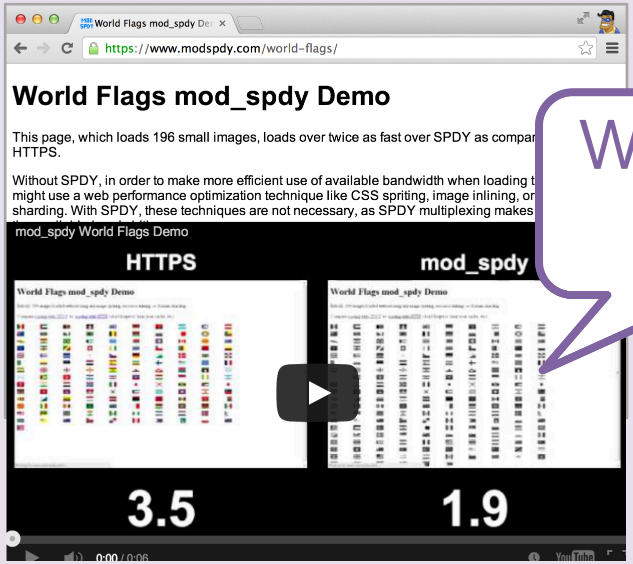
EQUAL

high

SPDY

Explanation

Unlike in HTTP, a TCP segment can carry multiple Web objects in SPDY.



World Flags mod_spdy De

https://www.modspdy.com/world-flags/

World Flags mod_spdy Demo

This page, which loads 196 small images, loads over twice as fast over SPDY as compar
HTTPS.

Without SPDY, in order to make more efficient use of available bandwidth when loading t
might use a web performance optimization technique like CSS spriting, image inlining, or
sharding. With SPDY, these techniques are not necessary, as SPDY multiplexing makes

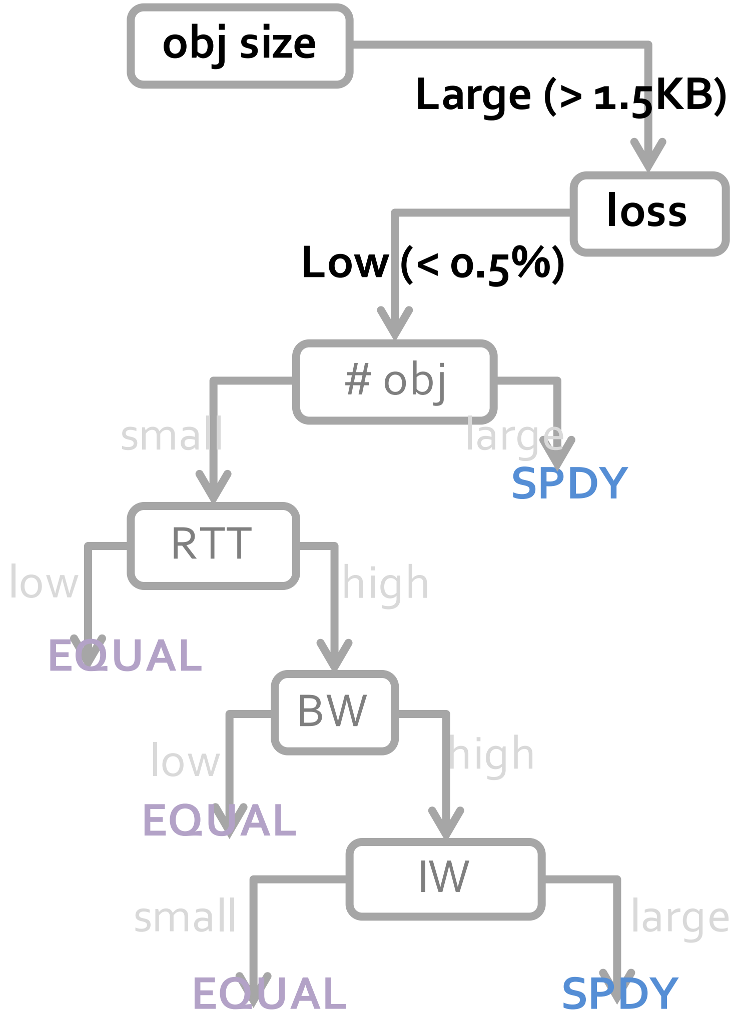
mod_spdy World Flags Demo

HTTPS	mod_spdy
3.5	1.9

0:00 / 0:06

Why SPDY helps

SPDY helps on large objects, low loss

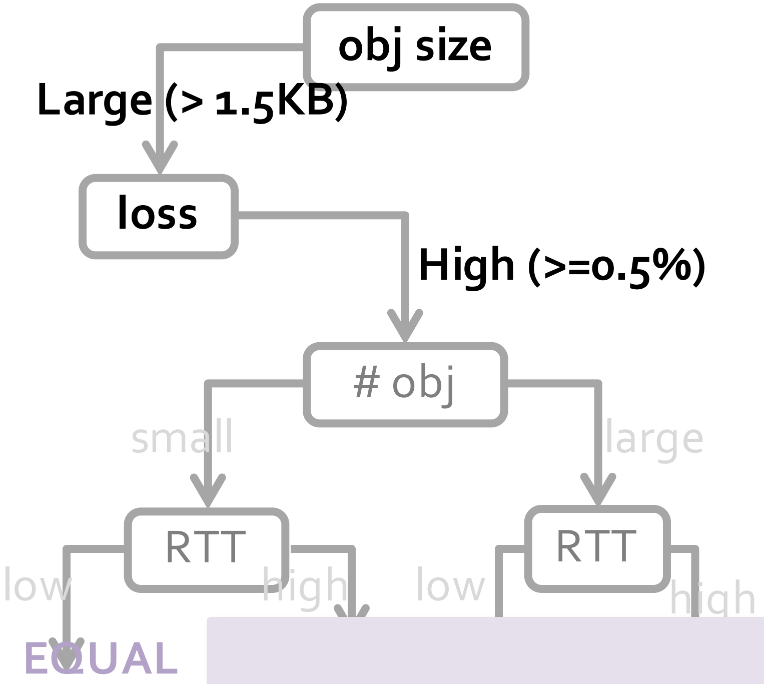


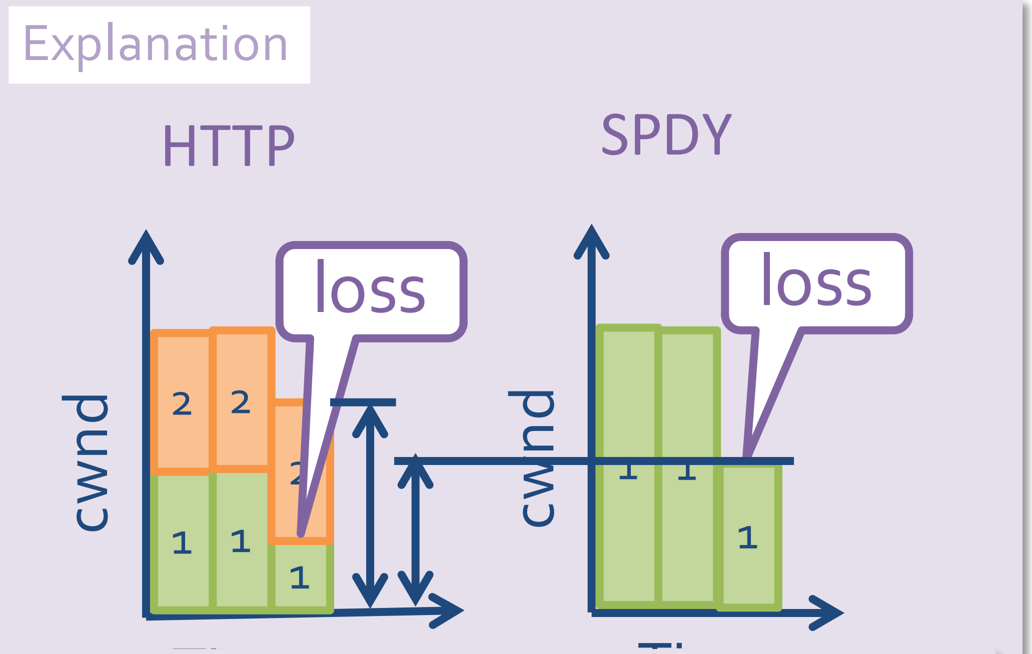
Explanation

In HTTP, Multiple connections compete with each other

→ More **retransmissions**

SPDY hurts on large objects, high loss





Most performance impact of SPDY comes from a single TCP connection.

Identify dominant factors

# obj

BW

RTT

Importance?

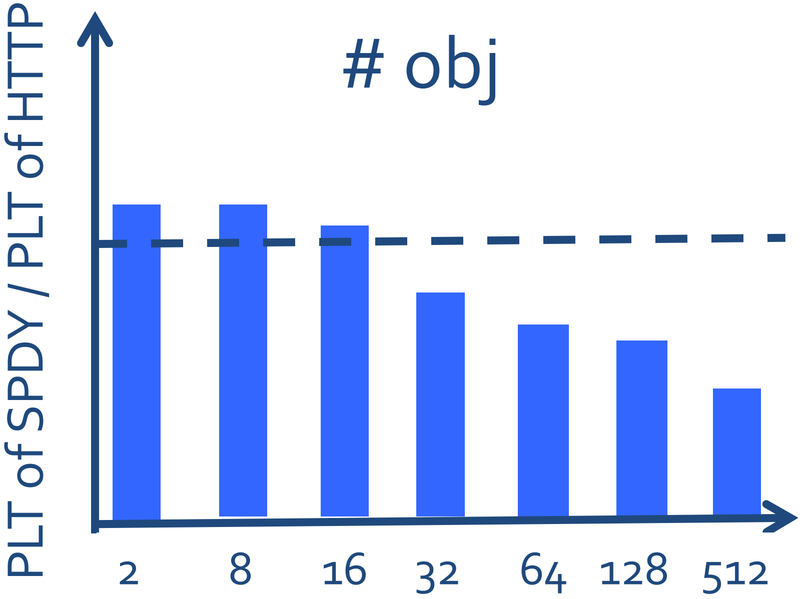
obj size

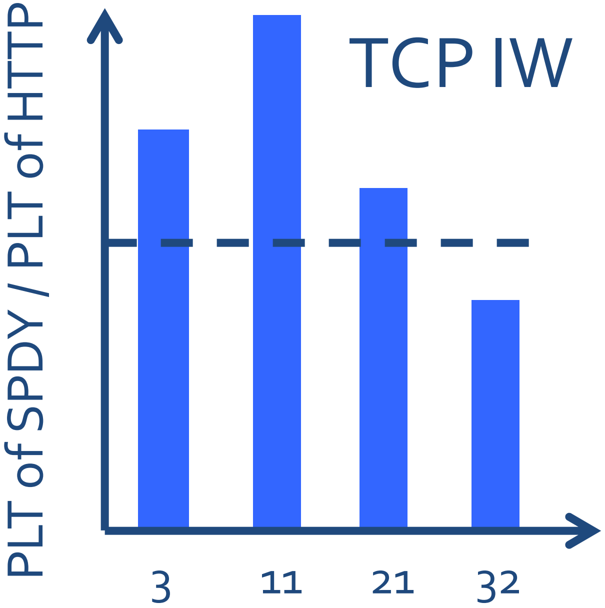
loss

IW

Identify dominant factors

obj size loss # obj more important than RTT BW IW





RTT: 200ms
BW: 10Mbps
Loss: 0
IW: 3
obj size: 10KB
# obj: 8

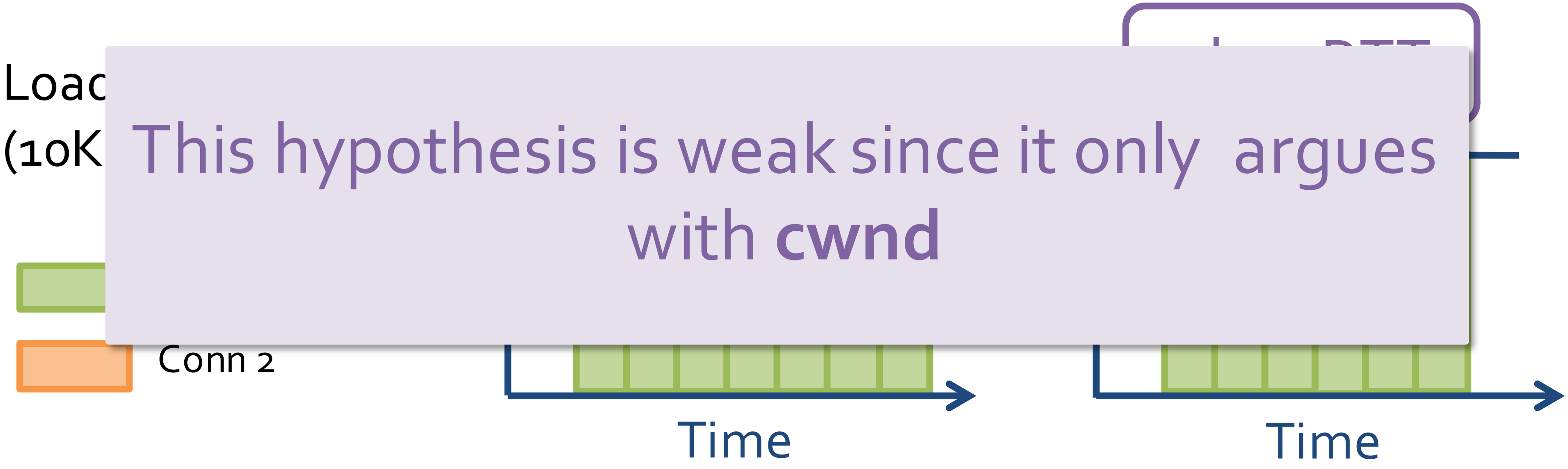
# obj shows a trend

IW doesn't show a trend

Does SPDY help stragglers?

HTTP/1.1

SPDY



This hypothesis is weak since it only argues with **cwnd**

- In our experiments, we find that SPDY helps little for stragglers.

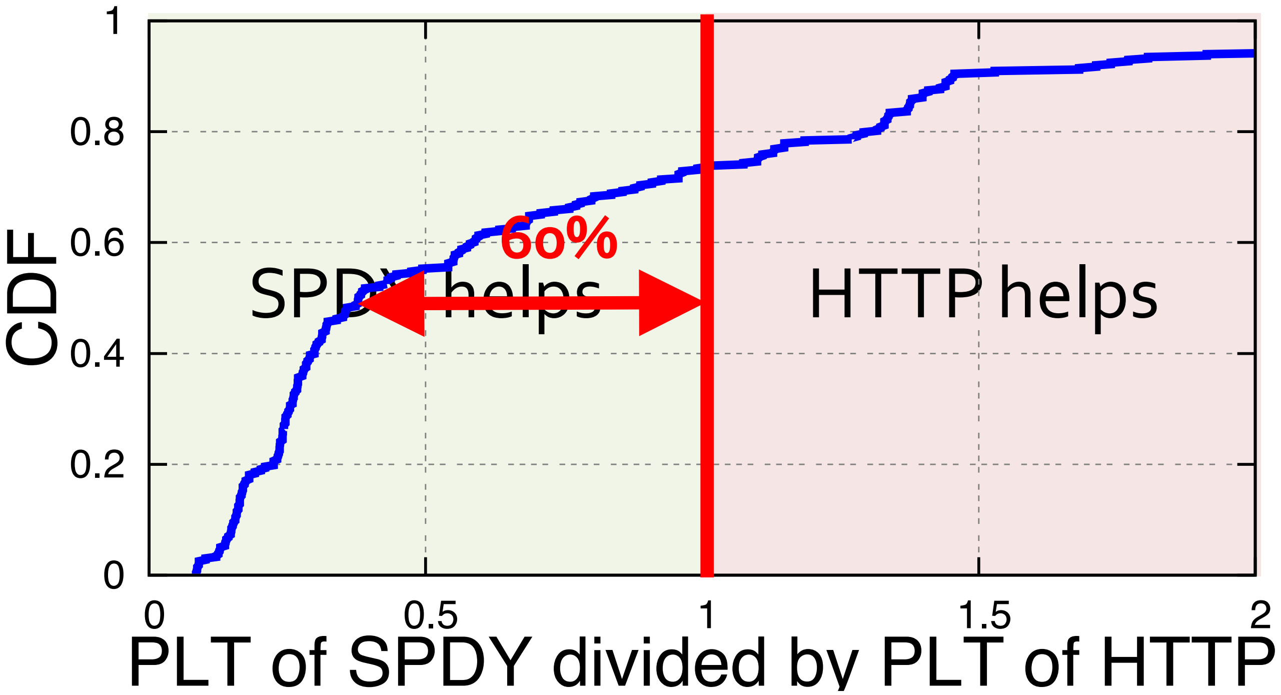
Outline

- Understanding SPDY's performance with
  - Synthetic objects
  - **Real objects**
  - Real pages
- Enhanced policies for SPDY

Synthetic objects → Real objects

	Factors	Range
Network parameters	RTT	20ms, 100ms, 200ms
	Bandwidth	1Mbps, 10Mbps
	Loss rate	0, .5%, 1%, 2%
TCP settings	TCP IW	3, 10, 21, 32
Web objects	Web obj. size	Top 200 Alexa pages
	# of objects	

Make HTTP requests



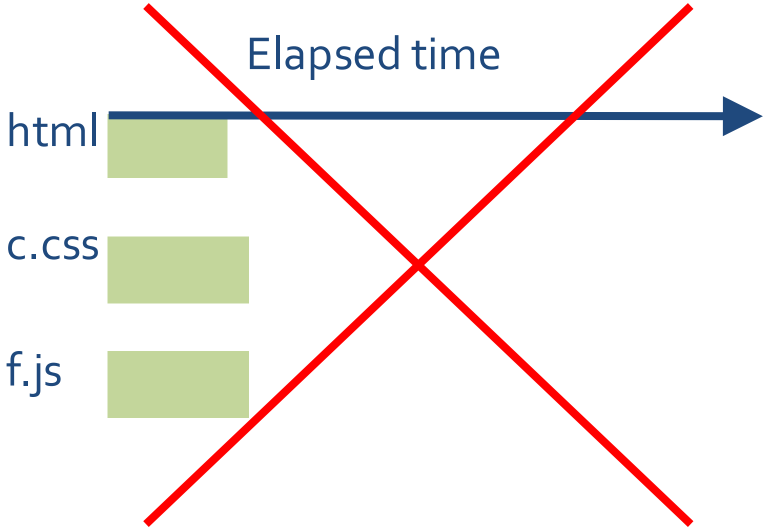
SPDY helps 60% in the median case because it largely reduces retransmissions

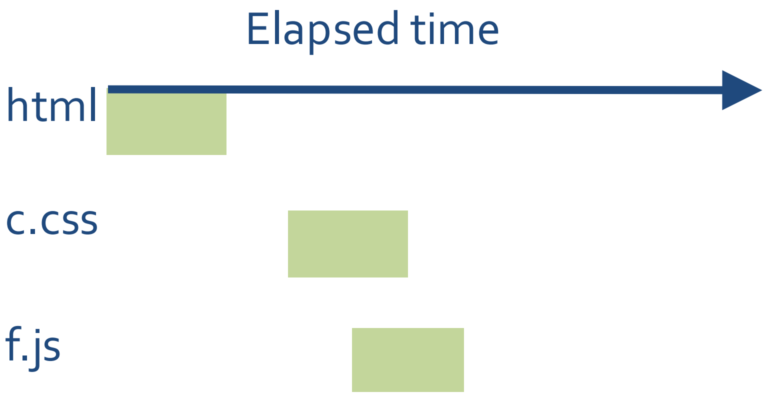
Outline

- Understanding SPDY’s performance with
  - Synthetic objects
  - Real objects
  - **Real pages**
- Enhanced polic DY

Browser effects

Browser effects

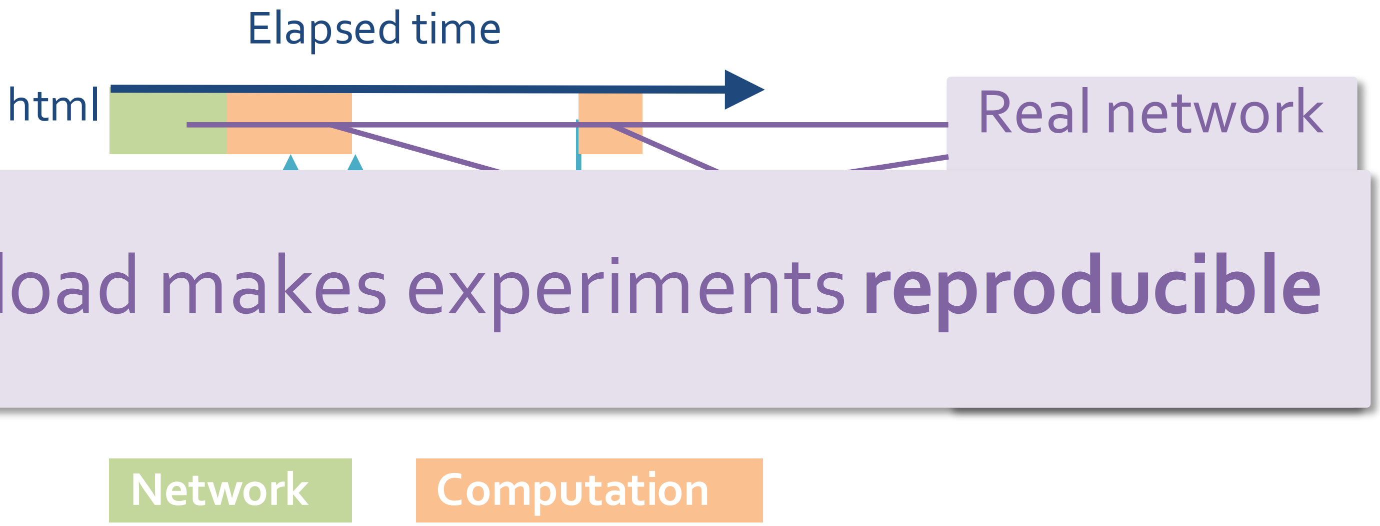




Assumption that objects are fetched at the same time does not hold.

Epload captures browser effects

- Recorder: capture the dependency graph
- Replayer: make network requests while simulating the computation portions

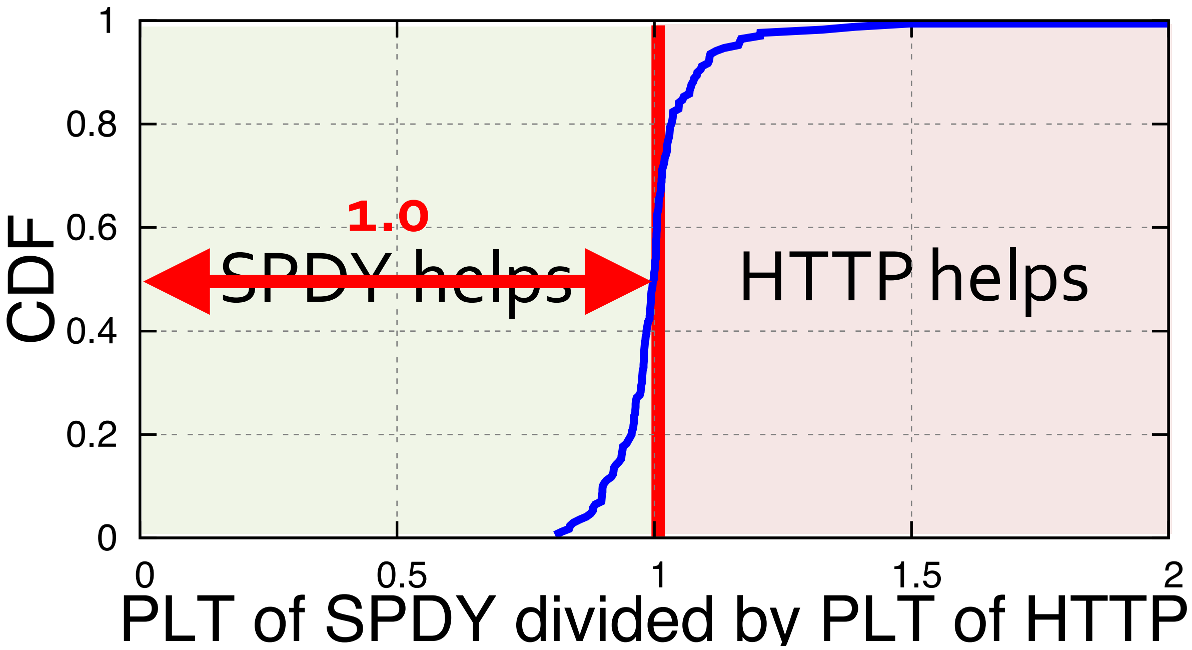


Epload makes experiments **reproducible**

Real objects → Real pages

	Factors	Range
Network parameters	RTT	20ms, 100ms, 200ms
	Bandwidth	1Mbps, 10Mbps
	Loss rate	0, .5%, 1%, 2%
TCP settings	TCP IW	3, 10, 21, 32
Web objects	Web obj. size	Top 200 Alexa pages
	# of objects	

Emulate page loads with **Epload**



SPDY helps marginally because

- Computation and dependencies

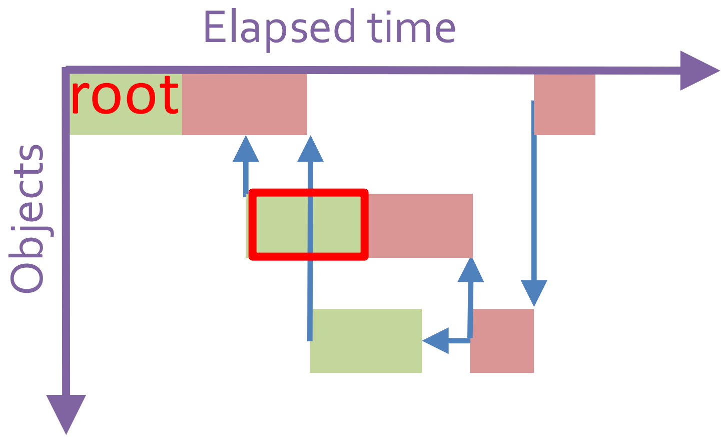
Dependencies and computation in real page loads reduce the impact of SPDY.

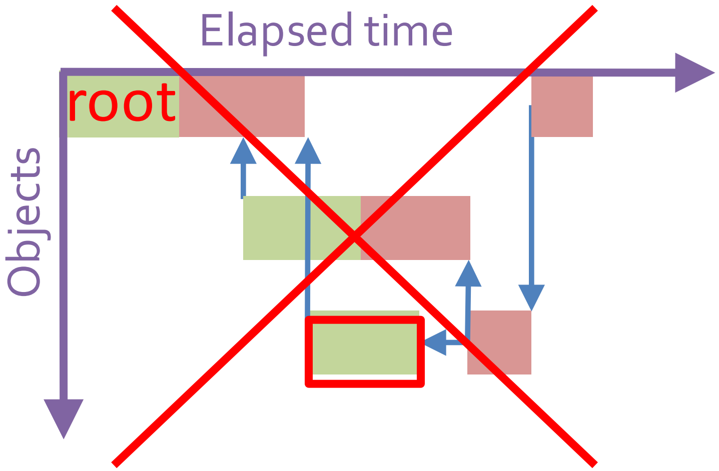
Outline

- Understanding SPDY's performance with
  - Synthetic objects
  - Real objects
  - **Real pages**
- Enhanced policies for SPDY

Improving SPDY with server push

- Leverage information from dependency graphs
  - Web objects that are closer to the root should be pushed earlier

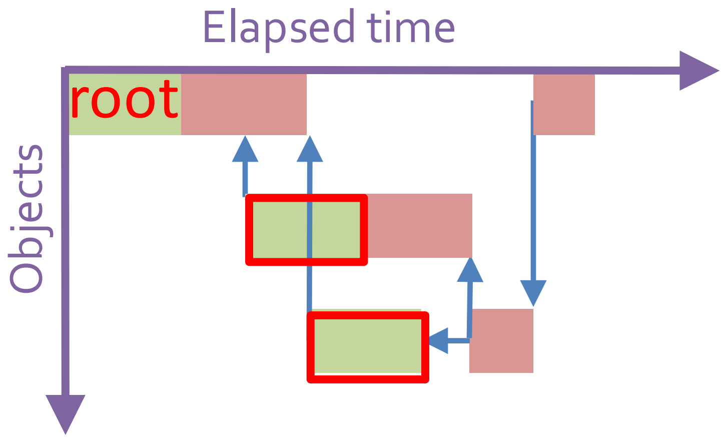




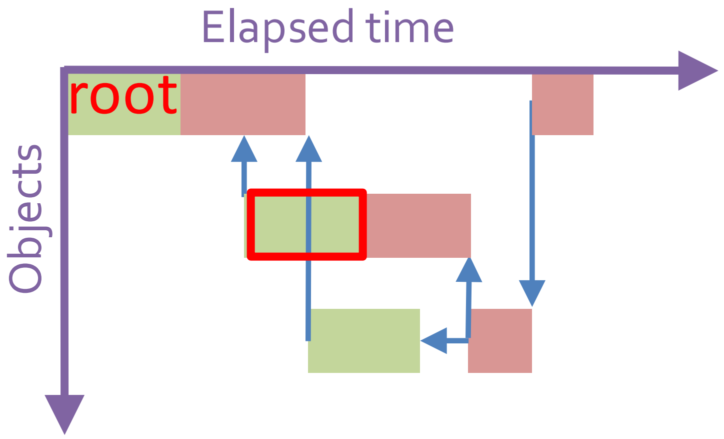
Improving SPDY with server push

Mod_spdy's: one level of HTML embeding

Our policy: one level of the dependency graph



Mod_spdy's



Our policy

Improving SPDY with server push

- Server push with our policy and mod_spdy's both helps page load time by **10%~30%**
- Our server push policy reduces **80%** of pushed bytes compared to mod_spdy's

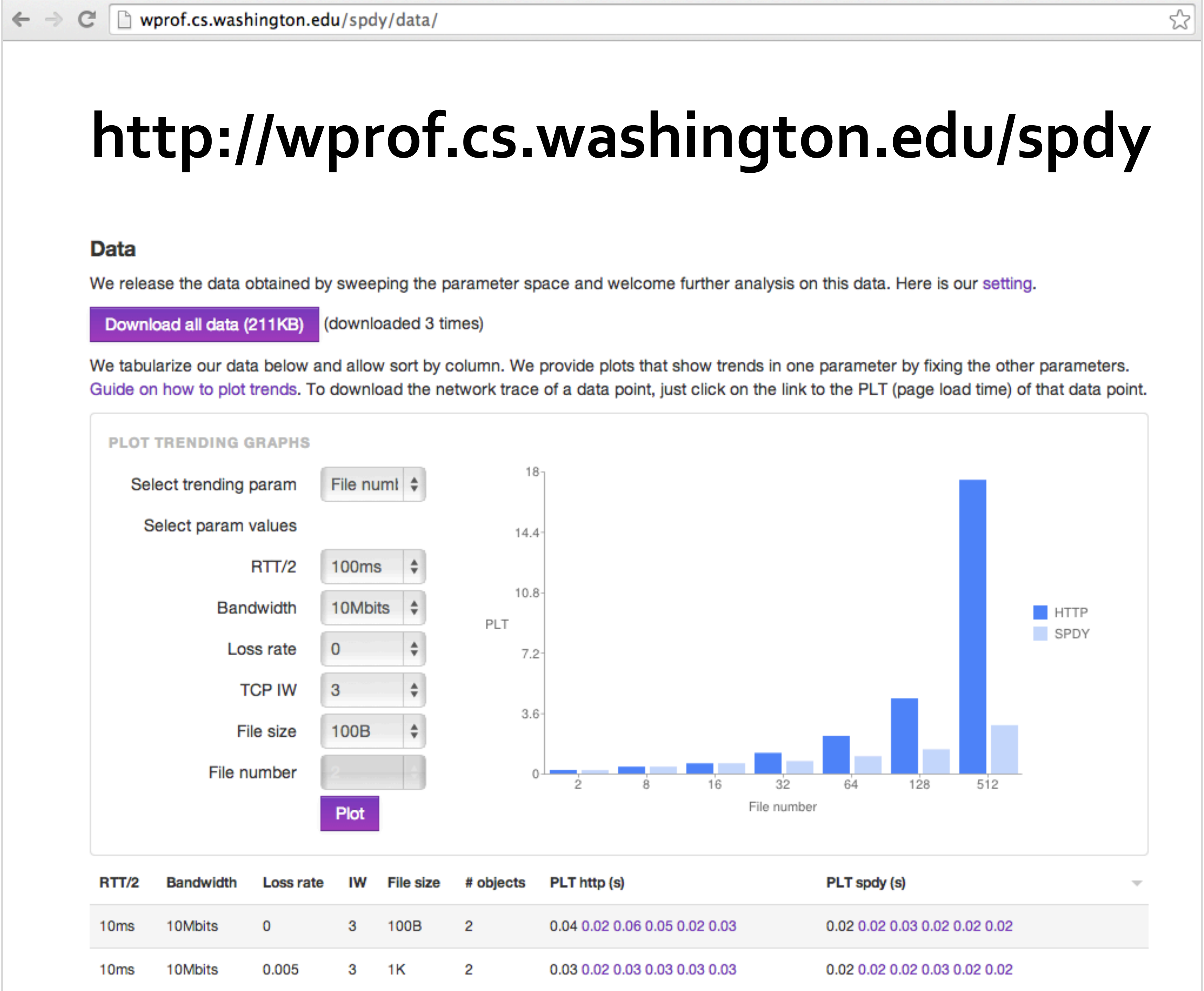
Improving page load performance requires restructuring the page load process, e.g. server push.

Other experiments in the paper

- With domain sharding
  - Compared domain sharding policies
    - One conn. for the whole page
    - One conn. for a second-level domain (SLD)
    - One conn. for a domain
  - Per-SLD policy is comparable to per-page; per-domain policy hurts performance
- With SSL/TLS
  - Tested SPDY and HTTP over SSL/TLS
  - Larger latencies but same conclusions

Conclusions

- We experimented with SPDY page loads over a large parameter space
- Most performance impact of SPDY over HTTP comes from its single TCP connection
- Browser computation and dependencies in real pages reduce the impact of SPDY
- To improve further, we need to restructure the page load process



http://wprof.cs.washington.edu/spdy

Data

We release the data obtained by sweeping the parameter space and welcome further analysis on this data. Here is our setting.

(downloaded 3 times)

We tabularize our data below and allow sort by column. We provide plots that show trends in one parameter by fixing the other parameters. Guide on how to plot trends. To download the network trace of a data point, just click on the link to the PLT (page load time) of that data point.

PLOT TRENDING GRAPHS

Select trending param: File numl

Select param values

RTT/2: 100ms

Bandwidth: 10Mbits

Loss rate: 0

TCP IW: 3

File size: 100B

File number: 2

RTT/2	Bandwidth	Loss rate	IW	File size	# objects	PLT http (s)	PLT spdy (s)
10ms	10Mbits	0	3	100B	2	0.04 0.02 0.06 0.05 0.02 0.03	0.02 0.02 0.03 0.02 0.02 0.02
10ms	10Mbits	0.005	3	1K	2	0.03 0.02 0.03 0.03 0.03 0.03	0.02 0.02 0.02 0.03 0.02 0.02