HTTP/2

One connection to rule them all

Jérémy Courtial @ Oodrive

Some background

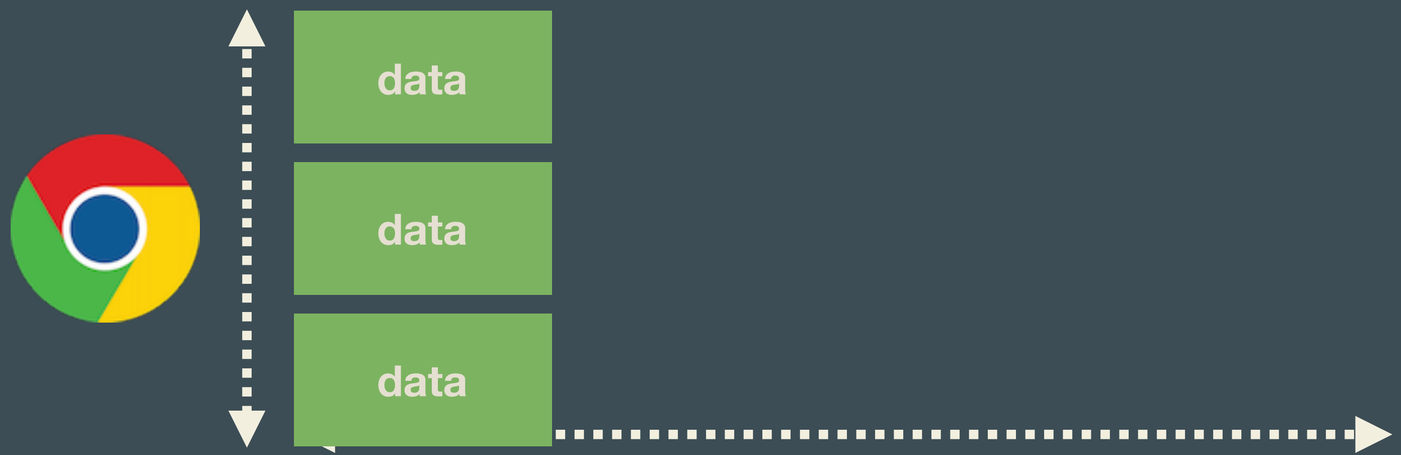
Latency

The time from the source sending a packet to the destination receiving it

Bandwidth

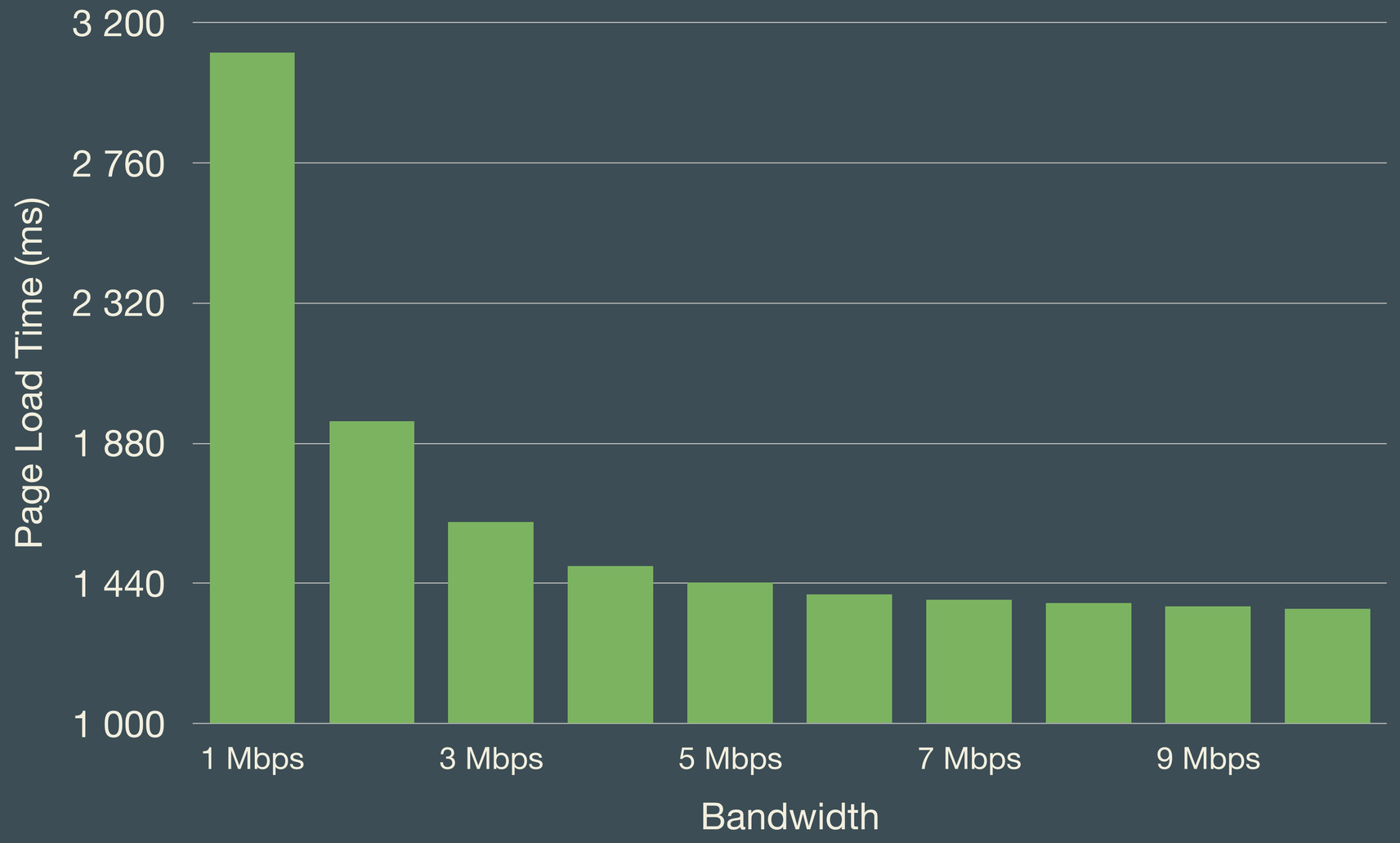
Maximum throughput of a logical or physical communication path

Bandwidth

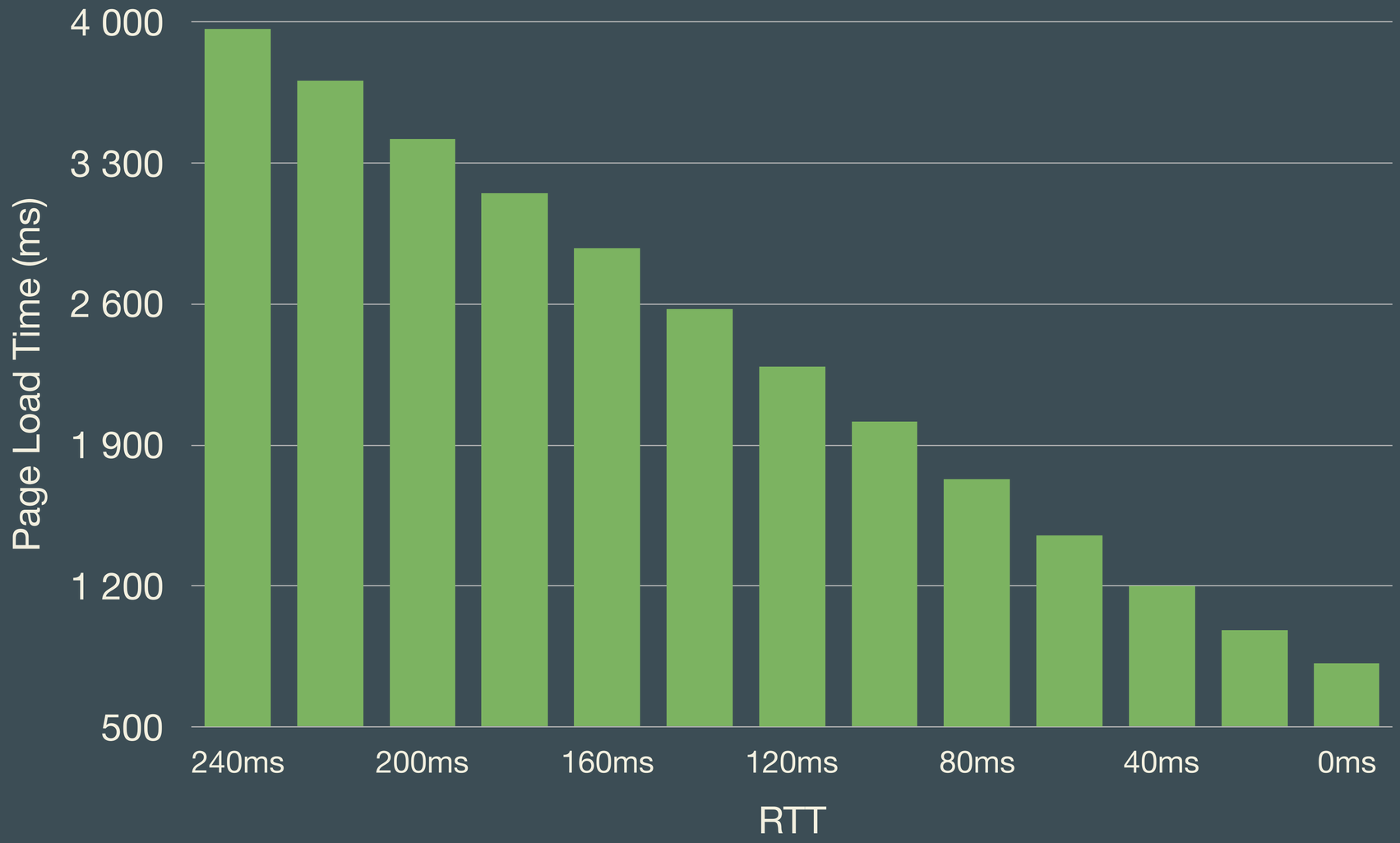


Latency

Impact of bandwidth



Impact of latency



Theory

NY - London through fiber

RTT ~60ms

3G network

RTT ~200-1000ms

Real life

ping www.nypdcadets.com

RTT ~150ms

ping agric.wa.gov.au

RTT ~340ms

Back to HTTP

Once upon a time,
in 1989, at the
CERN...

$> telnet cern.ch 80

Connected to 188.184.9.234

GET /hello

```
<html>
  <body>
    Hello Tim!
  </body>
</html>
```

Connection closed

GET /hello

<html>…

TCP SYN

TCP SYN/ACK

~100 ms

TCP ACK

GET /hello

<html>…

Closed

Entering cats ...

SYN

SYN/ACK

~100 ms

GET index.html

Closed

GET cat.jpg

SYN

SYN/ACK

~100 ms

So many more RTT

HTTP request
1-n RTTs

TLS handshake
1-2 RTTs

TCP handshake
1 RTT

DNS lookup
1 RTT

Keep-Alive

Because recycling is good

SYN

SYN/ACK

GET index.html

GET cat.jpg

Closed

Victory !

Yeah, sure. In 1996…

Modern web

Average requests/page

~115

Average size/page

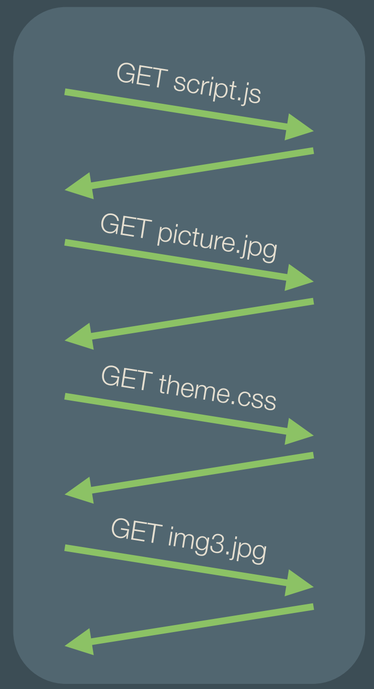
~2 500 kB

One active request per connection

Make a coffee

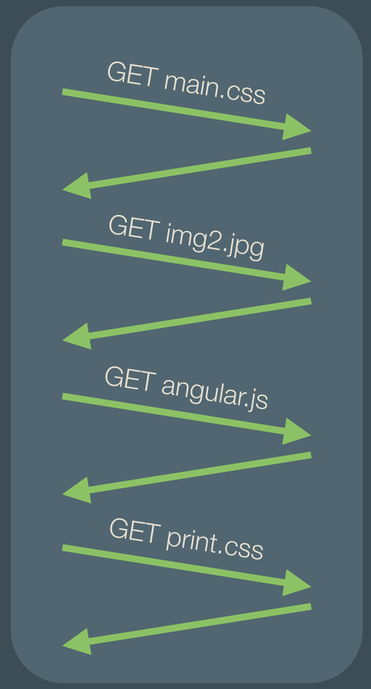
Pipelining ?

Head of
line blocking



Multiple connections

4-8 per hosts



Waterfall effect…

Optimizations…

- **Sharding** (network overhead)
- **Concatenation** (break cache)
- **Spriting** (break cache)
- **Inlining** (break cache, duplication)

Hack !

Hack !

Hack !

Hack !

Hack !

Introducing

HTTP/2

HTTP/2

Based on SPDY

Retains semantic compatibility with HTTP 1.x

Addresses « head of line blocking » & multiple connections issues

Improves overall performances

```
POST /hello
Host: domain.org
Content-Type: application/json
Content-Length: 12

{"greeting":"Hello world"}
```

HEADERS frame

```
POST /hello
Host: domain.org
Content-Type: application/json
Content-Length: 12
```

DATA frame

```
{"greeting":"Hello world"}
```

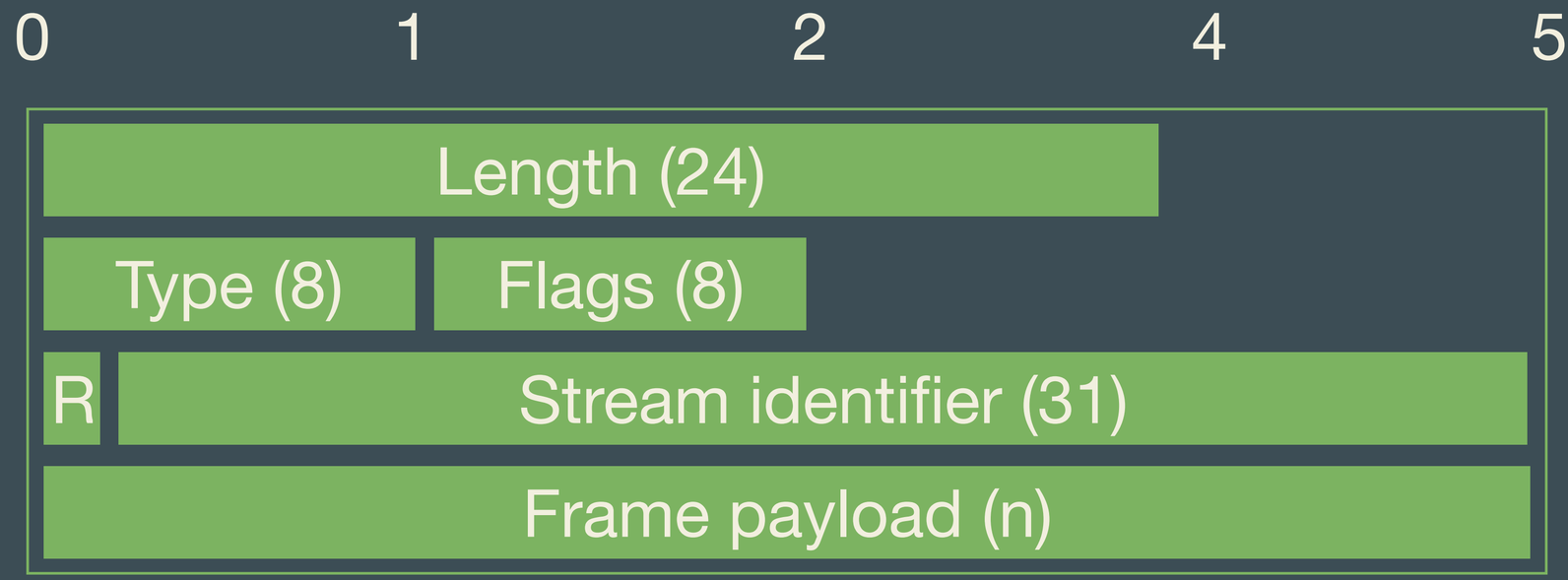
HEADERS frame

```
:authority: domain.org
:method: POST
:path: /hello
:scheme: https
content-type: application/json
content-length: 12
```

DATA frame

```
{"greeting":"Hello world"}
```

Binary format



16 Ko < max size < 16 Mo

SETTINGS

DATA

HEADERS

RST_STREAM

PRIORITY

Frame types

CONTINUATION

PING

PUSH_PROMISE

GOAWAY

WINDOW_UPDATE

Stream

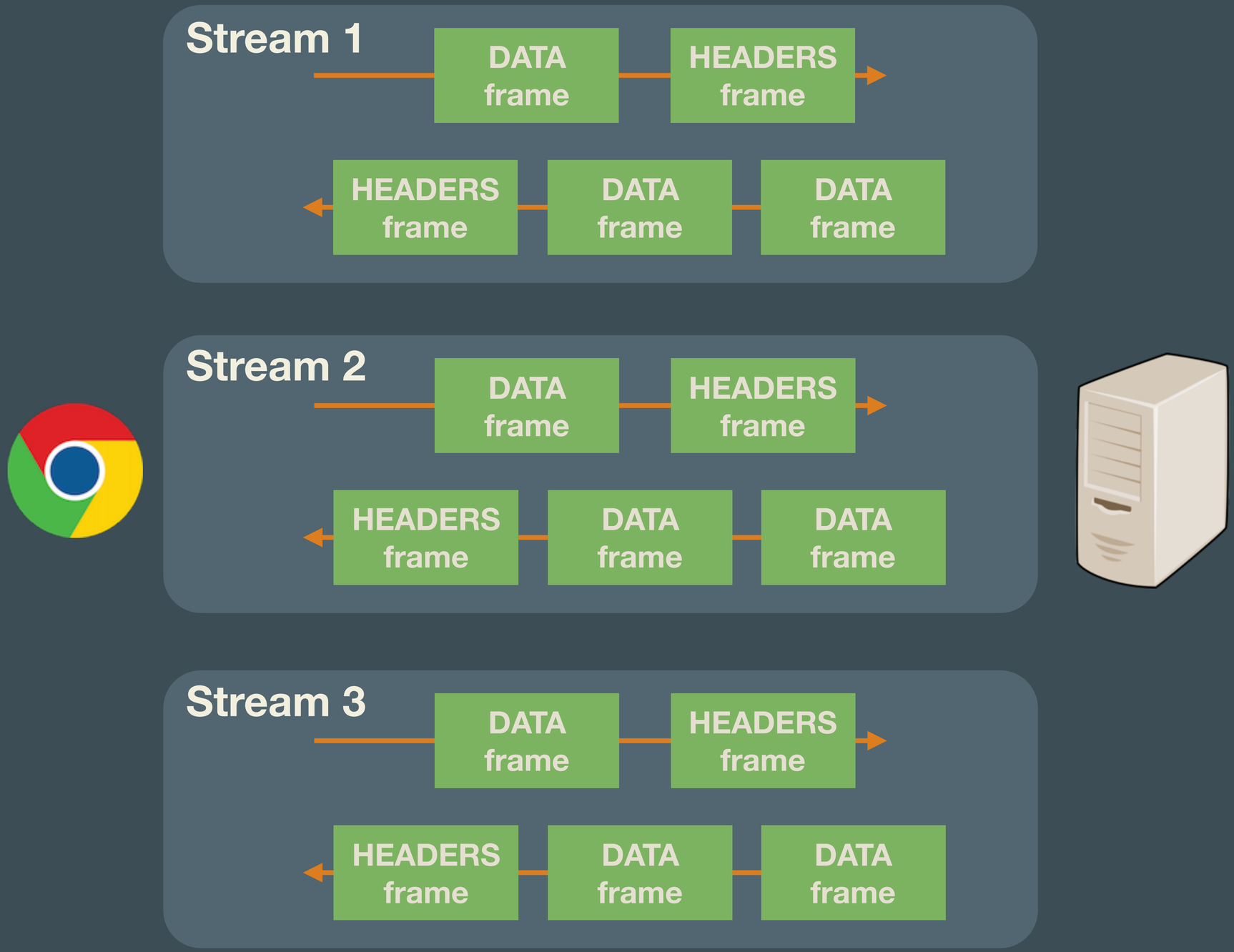
Request message

Response message

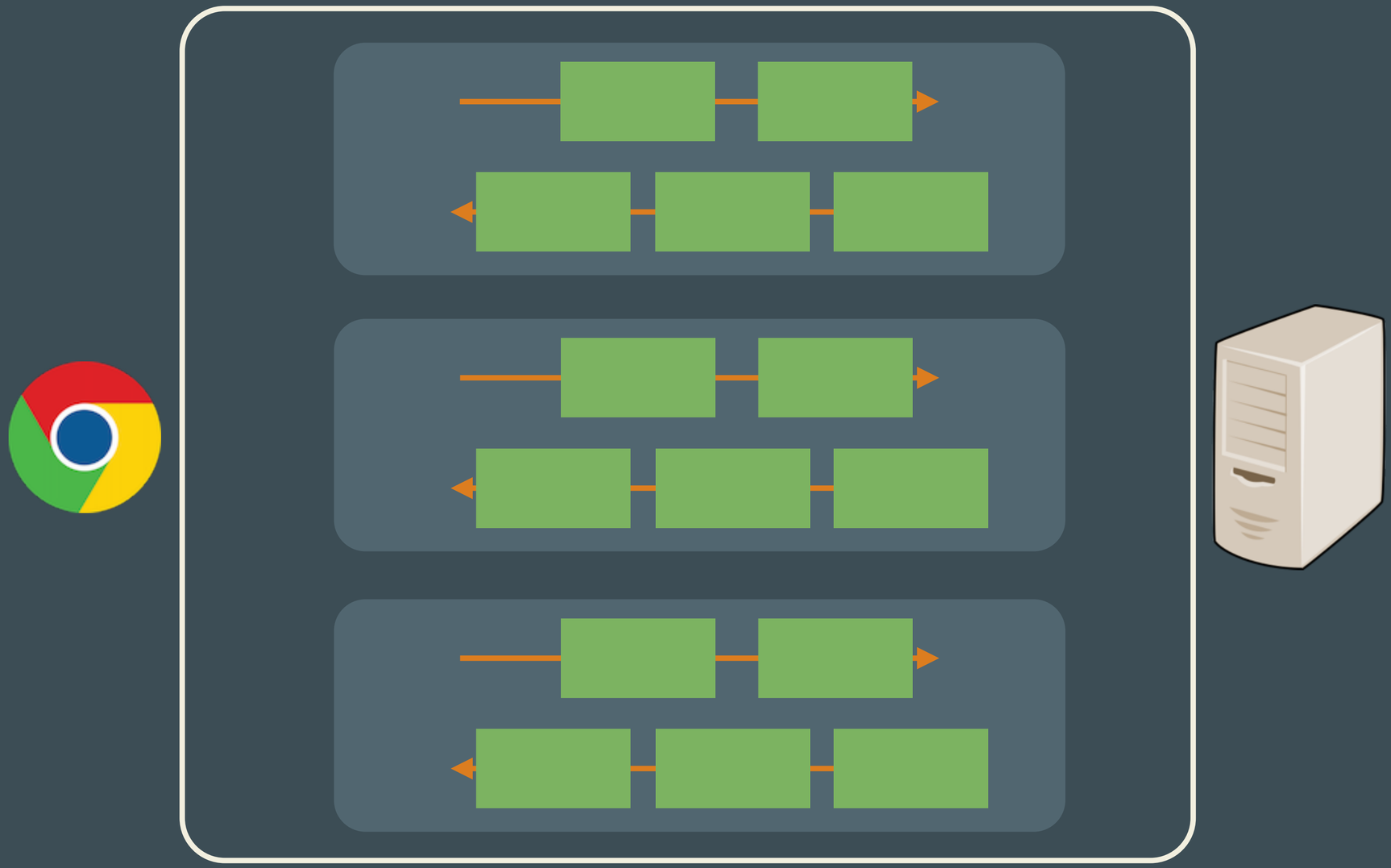
Stream

Request message

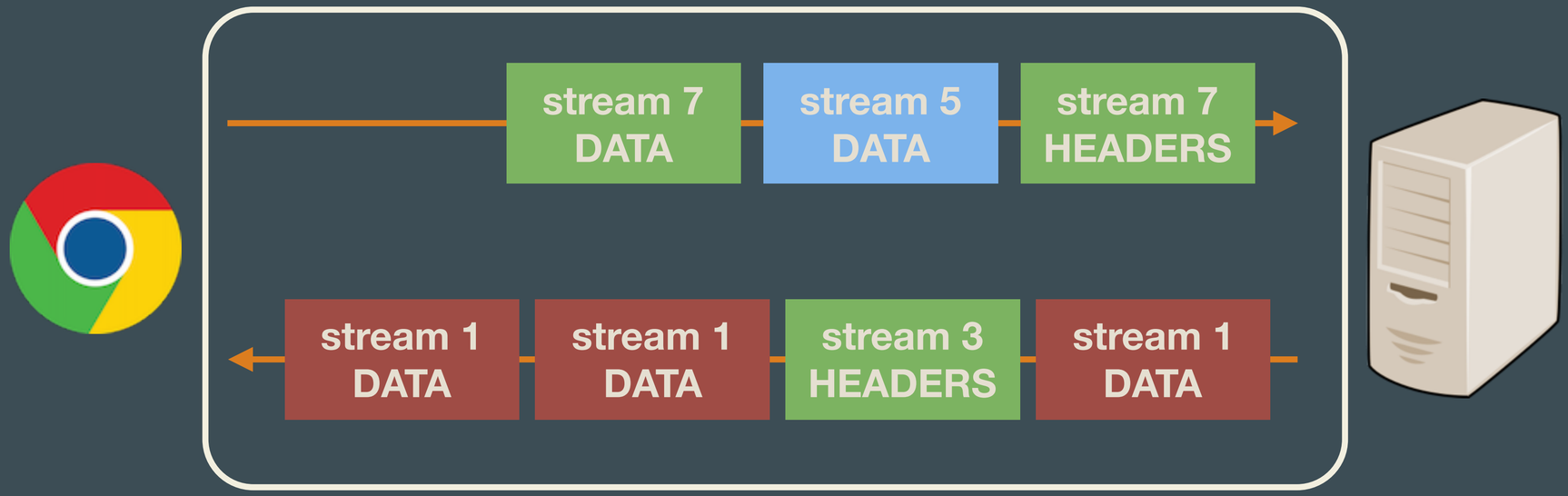
Response message


Stream 1
DATA frame
HEADERS frame
HEADERS frame
DATA frame
DATA frame
Stream 2
DATA frame
HEADERS frame
HEADERS frame
DATA frame
DATA frame
Stream 3
DATA frame
HEADERS frame
HEADERS frame
DATA frame
DATA frame

HTTP/2 connection

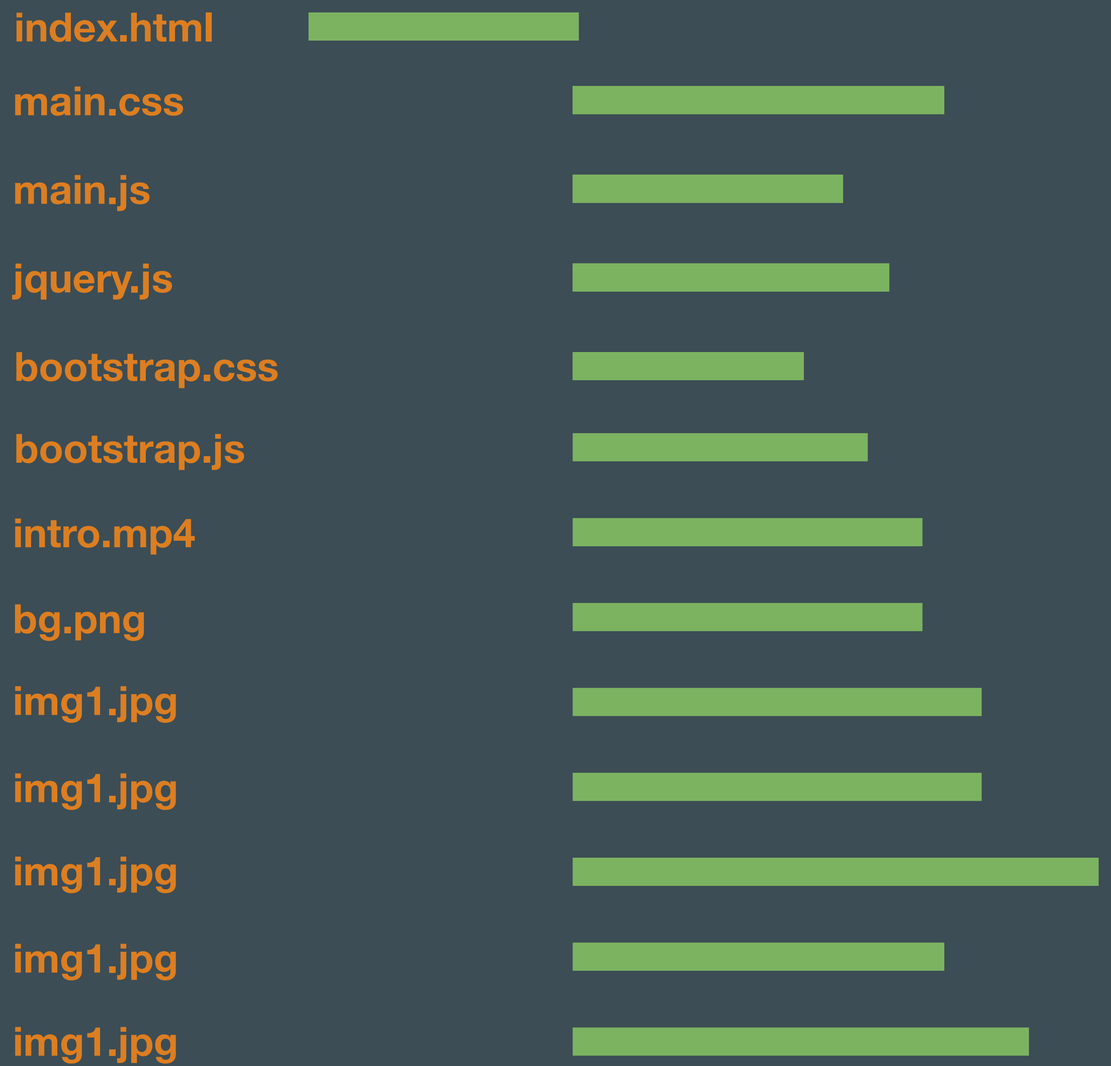


HTTP/2 connection



Remember that ?

Now more like that



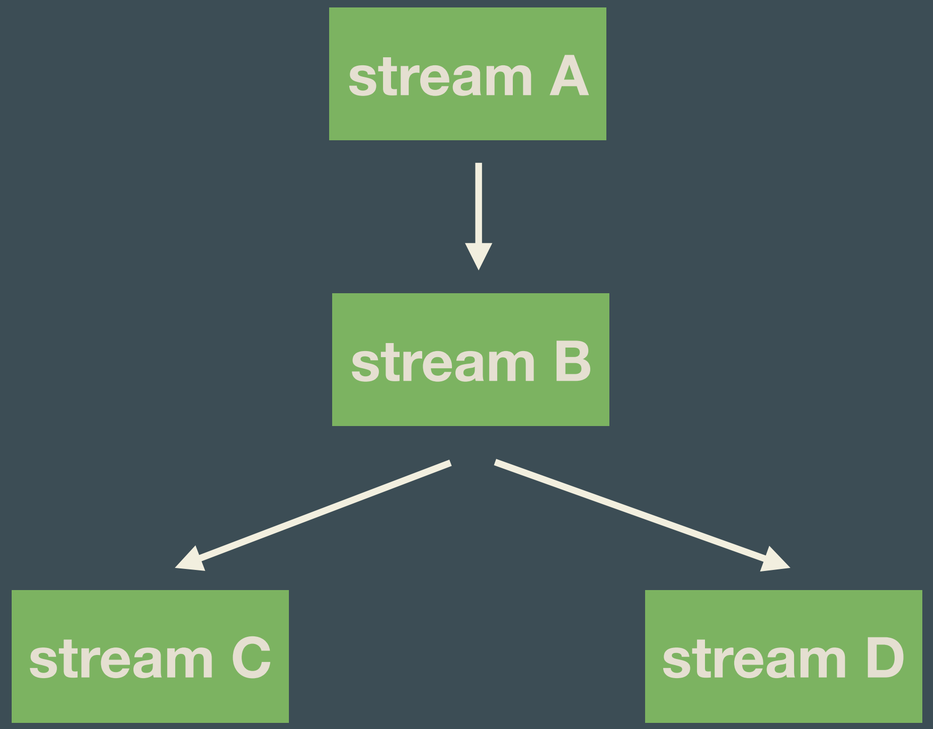
And there is more !

Stream Prioritization

HTML > CSS JS > Images

Or whatever algorithm you want !

Stream Dependency



Server push

Pushed resources can be …

cached

declined

disabled

prioritized

Take that inlining !

Headers compression

Request 1

method	GET
scheme	https
host	domain.org
path	/hello
user-agent	Mozilla/5.0

Request 2

method	GET
scheme	https
host	domain.org
path	/goodbye
user-agent	Mozilla/5.0

HEADERS frame (stream 1)

```
:method  GET
:scheme  https
:host  domain.org
:path  /hello
user-agent  Mozilla/5.0
```

Headers compression

Request 1

method	GET
scheme	https
host	domain.org
path	/hello
user-agent	Mozilla/5.0

HEADERS frame (stream 1)

```
:method  GET
:scheme  https
:host  domain.org
:path  /hello
user-agent  Mozilla/5.0
```

Request 2

method	GET
scheme	https
host	domain.org
path	/goodbye
user-agent	Mozilla/5.0

HEADERS frame (stream 2)

```
:path /goodbye
```

About those optimizations…

- ~~Sha~~
- ~~Co~~
- ~~Sprit~~
- ~~Inlining~~

Improvements

Google

Latency: -20%/-40%

Dropbox

Bandwidth usage: -50%

Ok, I’m in love !

But how do I switch ?

Upgrade & Discovery

Through TLS & ALPN

Because you do TLS, right ?

With prior knowledge

You just know

Upgrade header

Upgrade & Discovery

Through TLS & ALPN

Because you do TLS, right ?

~~With prior knowledge~~

~~You just know~~

~~Upgrade header~~

Are we there yet?

RFC 7540
May 2015

iOS

mod_http2

Jetty

Firefox

Safari

nginx

Implementations

IIS

IE (Win 10)

Netty

OkHttp

h2o

Chrome

F5

Deployment

2015 — Google

Twitter

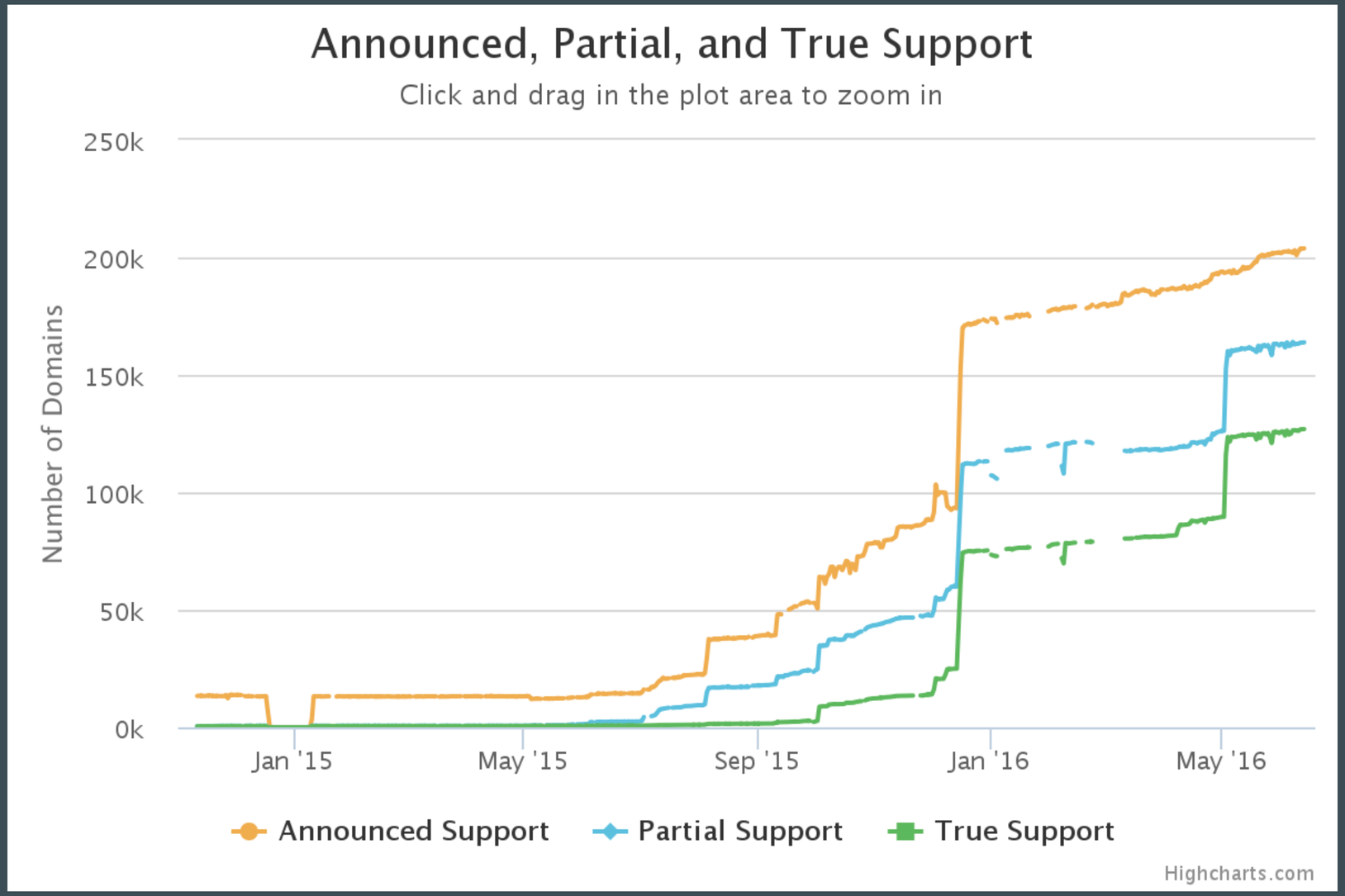
Wordpress

Akamai

CloudFlare

Blogspot

Today — Wikipedia


Announced, Partial, and True Support
Click and drag in the plot area to zoom in
250k
200k
150k
100k
50k
0k
Number of Domains
Jan '15
May '15
Sep '15
Jan '16
May '16
Announced Support
Partial Support
True Support
Highcharts.com

What's next ?

TCP Fast Open

TCP cookie for 0 RTT handshake

TLS 1.3

1-0 RTT handshakes

QUIC

HTTP over UDP

Bibliography

Ilya Grigorik (@igrigorik)

High Performance Browser Networking book

https://www.igvita.com/

CloudFlare blog posts

https://blog.cloudflare.com/tag/http2/

It's the Latency, Stupid

http://goo.gl/g26hOo , Stuart Cheshire

More Bandwidth Doesn't Matter (much)

http://goo.gl/wuvD1R , Mike Belshe

Thank you !

Questions ?