HTTP/2 Pros and Cons

Nick Shadrin / NGINX
nick@nginx.com
@shadrin
About me

• Nick Shadrin
• Technical Solutions Architect with NGINX
• Based in San Francisco
• 15 years experience with web tech
• nick@nginx.com / @shadrin
Links

All links in one page:

https://shadrin.org/talks/
Agenda

• Protocol overview
• HTTP/1 and HTTP/2 optimizations
• Troubleshooting
• Benchmarks
• Use of HTTP/2 with NGINX
• Conclusions
## HTTP History

<table>
<thead>
<tr>
<th>Year</th>
<th>Version</th>
<th>RFC#</th>
</tr>
</thead>
<tbody>
<tr>
<td>1991</td>
<td>0.9</td>
<td></td>
</tr>
<tr>
<td>1996</td>
<td>1.0</td>
<td>1945</td>
</tr>
<tr>
<td>1999</td>
<td>1.1</td>
<td>2616 and 7230 .. 7235</td>
</tr>
<tr>
<td>2015</td>
<td>2</td>
<td>7540</td>
</tr>
</tbody>
</table>

See Wikipedia
HTTP/1.1 example

GET /test HTTP/1.1
Host: example.com
User-Agent: Mozilla
X-Forwarded-For: 192.168.10.1
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate

HTTP/1.1 301 Moved Permanently
Server: nginx/1.9.9
Date: Tue, 19 Jan 2016 00:19:07 GMT
Content-Type: text/html
Content-Length: 184
Connection: close
Location: https://example.com/test
SPDY

- Announced in 2009 by Google
- Since then implemented in all major browsers
- Major goal: reduce page load time
- Major performance enhancements:
  - Compressed headers
  - Flow control
  - Server Push
HTTP/2 overview

- Introduced in 2015 as a standard
- Based on SPDY
- Includes major changes compared to HTTP/1:
  - Binary headers with HPACK
  - Multiple streams
  - Prioritization
  - Server Push
Encryption

• Is encryption required?
  • Spec says "No"
  • Browser vendors say "Yes"
Protocol negotiation

- Upgrade header
- NPN
- ALPN
NPN

- Next protocol negotiation
- Server lists the protocols
- Client picks one
ALPN / RFC7301

• Application level protocol negotiation
• Client lists the protocols
• Server picks one
• Results in fewer round trips
• Available in openssl 1.0.2
Revise your optimizations

- Domain sharding
- Image sprites
- Concatenating code files
Domain Sharding

• Browser opens 6 connections to the host

• Distribute your resources through multiple domains

• Does it help when you use HTTP/2? - No.
Image Sprites

- Aggregate multiple images in a single file
- Separate images on the client side

- Does it help when you use HTTP/2? - Somewhat.
Concatenating code files

- Combine JS and CSS into larger files

- Does it help when you use HTTP/2? - Not significantly.
Revise your optimizations

• Domain sharding
• Image sprites
• Concatenating code files

• All these optimizations add to the management overhead.
HTTP/2 stats today

• Is it time already to use HTTP/2? Let's look at the statistics
HTTP/2 protocol

Networking protocol for low-latency transport of content over the web. Originally started out from the SPDY protocol, now standardized as HTTP version 2.

<table>
<thead>
<tr>
<th>Browser</th>
<th>IE</th>
<th>Edge</th>
<th>Firefox</th>
<th>Chrome</th>
<th>Safari</th>
<th>Opera</th>
<th>iOS Safari</th>
<th>Opera Mini</th>
<th>Android Browser</th>
<th>Chrome for Android</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8</td>
<td>12</td>
<td>42</td>
<td>45</td>
<td>46</td>
<td>8.4</td>
<td>9</td>
<td>34</td>
<td>4.3</td>
<td>4.4</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.4.4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

HTTP/2 usage ([caniuse.com](http://caniuse.com))
HTTP/2 protocol (caniuse.com)

Networking protocol for low-latency transport of content over the web. Originally started out from the SPDY protocol, now standardized as HTTP version 2.
HTTP/2 usage (w3techs.com)
Downsides

- Greater overhead for the single connection.
- You might not need SSL.
- HTTP/1.x optimizations hurt.
- Big downloads don’t benefit.
- Your customers may not care.
HTTP/2 is unreadable
Troubleshooting HTTP/2 with Wireshark

1. Set the key log file ENV variable:
   export SSLKEYLOGFILE=/Users/path/file.key

2. Open your browser:
   open -a Google\ Chrome

3. Set the key file in Wireshark
Wireshark · Preferences

Secure Sockets Layer

RSA keys list [Edit...]

SSL debug file

/Users/nick/Documents/ssl/wire.dbg [Browse...]

- Reassemble SSL records spanning multiple TCP segments
- Reassemble SSL Application Data spanning multiple SSL records
- Message Authentication Code (MAC), ignore "mac failed"

Pre-Shared-Key

(Pre)-Master-Secret log filename

/Users/nick/key.key [Browse...]

Help [Cancel] [OK]
<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source (IP)</th>
<th>Destination (IP)</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>0</td>
<td>10.10.20.105</td>
<td>216.58.193.78</td>
<td>HTTP2</td>
<td>119</td>
<td>Magic</td>
</tr>
<tr>
<td>33</td>
<td>0</td>
<td>10.10.20.105</td>
<td>216.58.193.78</td>
<td>HTTP2</td>
<td>116</td>
<td>SETTINGS</td>
</tr>
<tr>
<td>34</td>
<td>0</td>
<td>10.10.20.105</td>
<td>216.58.193.78</td>
<td>HTTP2</td>
<td>108</td>
<td>WINDOW_UPDATE</td>
</tr>
<tr>
<td>35</td>
<td>0</td>
<td>10.10.20.105</td>
<td>216.58.193.78</td>
<td>HTTP2</td>
<td>503</td>
<td>HEADERS</td>
</tr>
<tr>
<td>37</td>
<td>0</td>
<td>216.58.193.78</td>
<td>10.10.20.105</td>
<td>HTTP2</td>
<td>122</td>
<td>SETTINGS</td>
</tr>
<tr>
<td>38</td>
<td>0</td>
<td>216.58.193.78</td>
<td>10.10.20.105</td>
<td>HTTP2</td>
<td>108</td>
<td>WINDOW_UPDATE</td>
</tr>
<tr>
<td>42</td>
<td>0</td>
<td>10.10.20.105</td>
<td>216.58.193.78</td>
<td>HTTP2</td>
<td>104</td>
<td>SETTINGS</td>
</tr>
<tr>
<td>44</td>
<td>0</td>
<td>10.10.20.105</td>
<td>216.58.193.78</td>
<td>HTTP2</td>
<td>104</td>
<td>SETTINGS</td>
</tr>
<tr>
<td>46</td>
<td>0</td>
<td>216.58.193.78</td>
<td>10.10.20.105</td>
<td>HTTP2</td>
<td>473</td>
<td>HEADERS</td>
</tr>
<tr>
<td>47</td>
<td>0</td>
<td>216.58.193.78</td>
<td>10.10.20.105</td>
<td>HTTP2</td>
<td>104</td>
<td>DATA</td>
</tr>
<tr>
<td>48</td>
<td>0</td>
<td>216.58.193.78</td>
<td>10.10.20.105</td>
<td>HTTP2</td>
<td>112</td>
<td>PING</td>
</tr>
<tr>
<td>52</td>
<td>0</td>
<td>10.10.20.105</td>
<td>216.58.193.78</td>
<td>HTTP2</td>
<td>112</td>
<td>PING</td>
</tr>
</tbody>
</table>

```
[Weight real: 183]
Header Block Fragment: 0085b9495339e4074f5054494f4e53418bae81fa5e639e6a...
[Header Length: 665]
  ▶ Header: :method: OPTIONS
  ▶ Header: :authority: play.google.com
  ▶ Header: :scheme: https
  ▶ Header: :path: /log?format=json
  ▶ Header: access-control-request-method: POST
  ▶ Header: origin: https://www.google.com
  ▶ Header: user-agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome...
  ▶ Header: access-control-request-headers: accept, authorization, content-type, x-goog-authuser
  ▶ Header: accept: */*
  ▶ Header: x-client-data: CKW2yQEixLbJAg91c0B
  ▶ Header: dnt: 1
  ▶ Header: referer: https://www.google.com/_/chrome/newtab-serviceworker.js
  ▶ Header: accept-encoding: gzip, deflate, sdch
```

Frame (503 bytes)  Decrypted SSL data (408 bytes)  Decompressed Header (665 bytes)
**HyperText Transfer Protocol 2**

Stream: HEADERS, Stream ID: 3, Length 131

- **Length:** 131
- **Type:** HEADERS (1)
- **Flags:** 0x04
  - .... .0 = End Stream: False
  - .... .1.. = End Headers: True
  - .... 0... = Padded: False
  - ..0. .... = Priority: False
  - 00.0 ..0. = Unused: 0x00

0... .... ..... ..... ..... ..... .... = Reserved: 0x00000000

.000 0000 0000 0000 0000 0000 0000 0000 0011 = Stream Identifier: 3

[Pad Length: 0]

Header Block Fragment: 88760b6e67696e782f312e392e34611d4d6f6e2c20313820...

[Header Length: 217]

- **Header:** :status: 200
- **Header:** server: nginx/1.9.4
- **Header:** date: Mon, 18 Jan 2016 23:51:37 GMT
- **Header:** content-type: text/css
- **Header:** last-modified: Sun, 19 Jul 2015 18:00:27 GMT
- **Header:** etag: W/"55abe5bb-df"
- **Header:** content-encoding: gzip

Padding: <MISSING>
Benchmarks
Benchmarks from nginx.conf 2015

Test Environment

Hardware: Intel Core i7-4770S, 16Gb of RAM, no disk I/O was involved
Kernel: Linux 4.0.9-gentoo
Network: loopback, 1400 MTU, netem
Server: nginx 1.9.5
Client: Chromium 45.0.2454.85 (64-bit) via Selenium WebDriver

The results were analysed using ministat: http://www.freebsd.org/cgi/man.cgi?query=ministat

Please note, that absolute numbers are irrelevant. Look at the trend.

nginx.conf

events {}

http {
  include conf/mime.types;
  root ..:/http2rulez.com/public;
  ssl_certificate ..:/server-ca.crt;
  ssl_certificate_key ..:/server.key;
  ssl_buffer_size 4k;
  output_buffers 2 1m;
  gzip on;
  gzip_types text/css application/javascript;

  server {
    listen 127.0.0.2:4433 ssl http2;
    location / { }
  }
  server {
    listen 127.0.0.2:4432 ssl;
    location / { }
  }
  server {
    listen 127.0.0.2:8080;
    location / { }
  }
}
Benchmark setup

- NGINX 1.9.9
- Ubuntu 15.10
- Openssl 1.0.2
- Chrome
- Measuring full page reload
New benchmark page

• Free template BCORE Admin: http://binarytheme.com/bootstrap-free-admin-dashboard-template/

• Added more images: 54 objects total
NGINX support for h2

user@server$ ./configure --with-http_v2_module --with-http_ssl_module

[...]
user@server$ nginx -V

nginx version: nginx/1.9.9
built by gcc 4.8.4 (Ubuntu 4.8.4-2ubuntu1~14.04)
built with OpenSSL 1.0.1f 6 Jan 2014
TLS SNI support enabled
configure arguments: --with-http_v2_module --with-http_ssl_module
NGINX config for h2

server {
    listen 443 ssl http2;
    server_name .example.com;
    ssl_certificate /etc/nginx/ssl/example.com.crt;
    ssl_certificate_key /etc/nginx/ssl/example.com.key;
    ssl_protocols TLSv1.2;
    root /data/example.com;
    location / {
        proxy_pass http://backend.example.com/;
    }
}

NGINX logs for h2

- $request = GET /url HTTP/2.0

- 10.2.2.2 - - [18/Jan/2016:16:51:40 -0800] "GET / HTTP/2.0" 200 3470 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.111 Safari/537.36"
NGINX Amplify for monitoring
Useful tools

caniuse.com
letsencrypt.com
webpagetest.org
Links

All links in one page:

https://shadrin.org/talks/