HTTP/2 y HTTP/3

Redes y Comunicaciones (2024)

Qué es HTTP/2?

- Reemplazo de cómo HTTP se transporta.
- No es un reemplazo del protocolo completo.
- Se conservan métodos y semántica.
- Base del trabajo protocolo desarrollado por Google SPDY/2.
- Definido en:
  - RFC7540: Hypertext Transfer Protocol version 2.
  - RFC7541: HPACK - Header Compression for HTTP/2.

Problemas con HTTP/1.0, HTTP/1.1

- Un request por conexión, por vez, muy lento.
- Alternativas (evitar HOL/HOLB Head of Line Blocking):
  - Conexiones persistentes y pipelining.
  - Generar conexiones paralelas.

- Problemas:
  - Pipelining requiere que los responses sean enviado en el orden solicitado, HOLB posible.
  - POST no siempre pueden ser enviados en pipelining.
  - Demasiadas conexiones genera problemas, control de congestión, mal uso de la red.
  - Muchos requests, muchos datos duplicados (headers).

Diferencias principales con HTTP/1.1

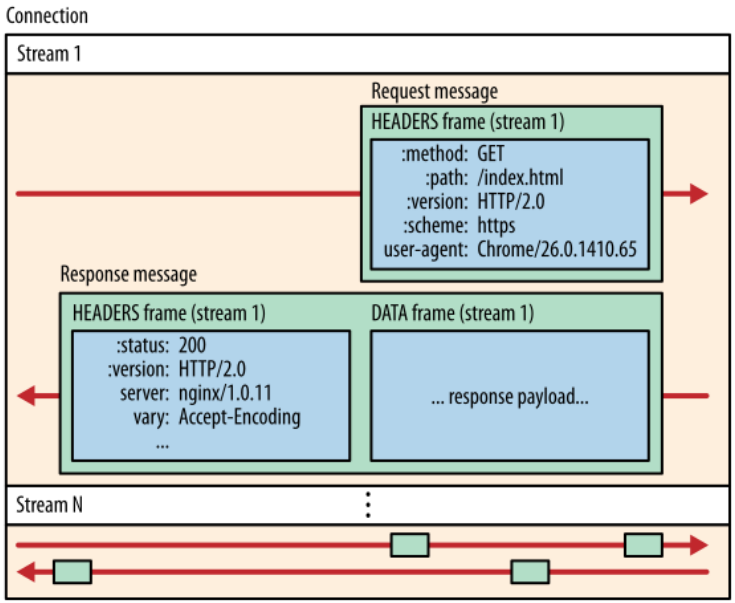
- Protocolo binario en lugar de textual(ASCII), binary framing: (más eficiente).
- Multiplexa varios request en una petición en lugar de ser una secuencia ordenada y bloqueante.
- Utilizar una conexión para pedir/traer datos en paralelos, agrega: datos fuera de orden, priorización, flow control por frame.
- Usa compresión de encabezado.
- Permite a los servidores “pushear” datos a los clientes.
- La mayoría de las implementaciones requieren TLS/SSL, no el estándar.

HTTP/2 mux stream, framing

- Puede generar una o más conexiones TCP. Trata de aprovechar las que tiene establecidas.
- Un stream es como una sub-conexión (una “conexión” http2 dentro de una conexión TCP).
- Un stream tiene un ID y una prioridad(alternativa) y son bidireccionales.
- Sobre una conexión TCP multiplexa uno o más streams (“conexiones http2”).
- Los streams transportan mensajes.
- Los mensajes http2 (Request, Response) se envían usando un stream.
- Los mensajes http2 son divididos en frames dentro del mismo stream.
- Un frame es una porción de mensaje: header fijo+payload variable (unidad mínima).
- El mismo stream puede ser usado para llevar diferentes msj.

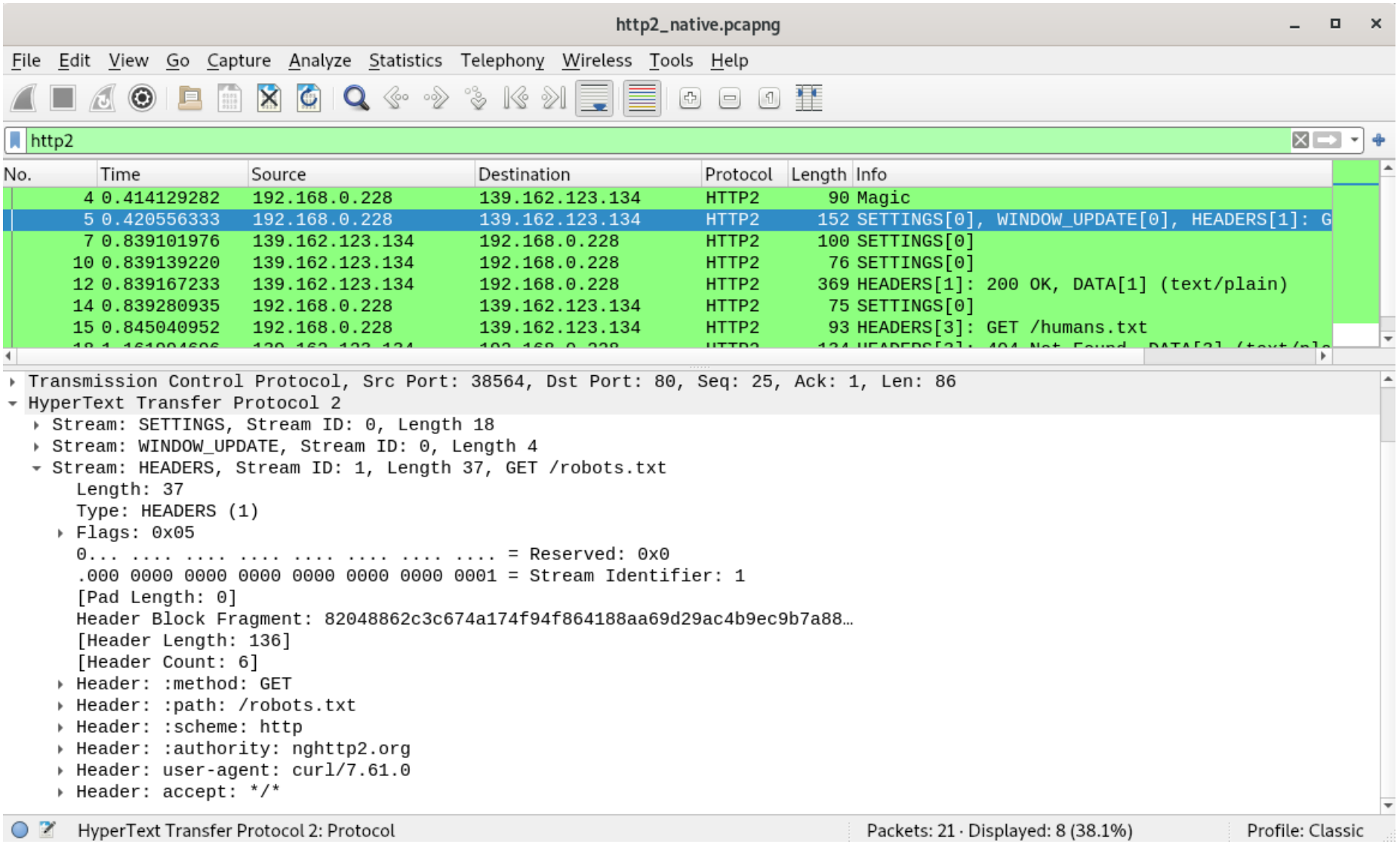
HTTP/2 mux stream, framing (cont.)

- Los mensajes están compuestos por frames, que podrían ser de diferentes tipos.
- Los streams van en una misma conexión.
- Los streams son identificados y divididos en frames.
- Frame types: HEADERS, DATA, PUSH_PROMISE, WINDOW_UPDATE, SETTINGS, etc.

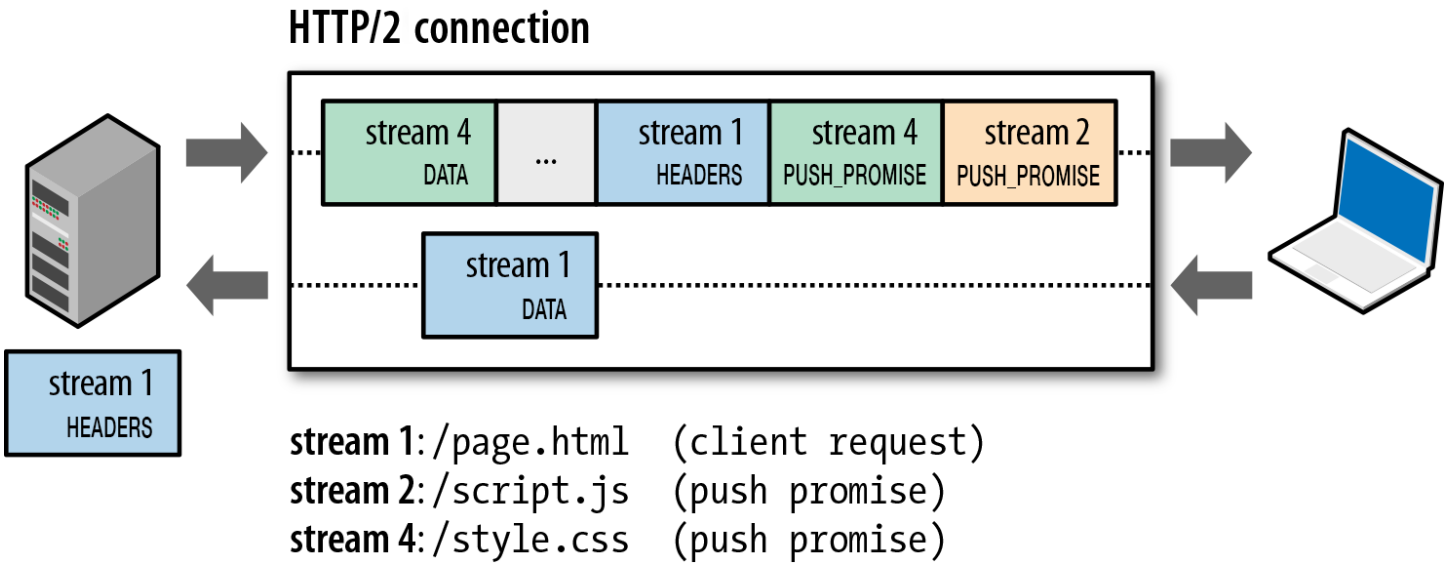


fuente: https://web.dev/performance-http2/#streams,-messages,-and-frames

HTTP/2 mux stream, framing (cont.)



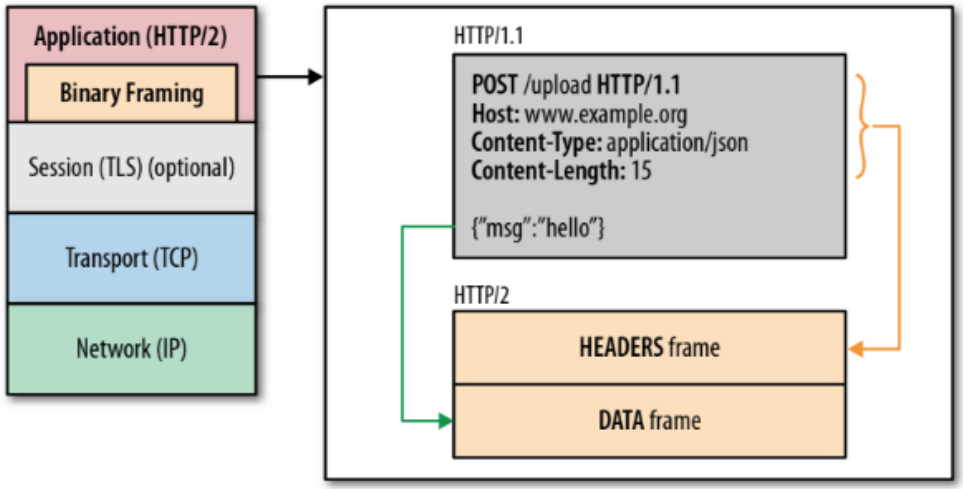
HTTP/2 mux stream, framing (cont.)



fuente: https://docs.google.com/presentation/d/1r7QXGYOLCh4fcUq0jDdDwKJWNqWK1o4xMtYpKZCJYjM/present?slide=id.p19

HTTP/2 mux stream, framing (Cont.)

- Streams codificados en binario y cada frame con header común fijo(9B).



fuente: https://docs.google.com/presentation/d/1r7QXGYOLCh4fcUq0jDdDwKJWNqWK1o4xMtYpKZCJYjM/present?slide=id.p19

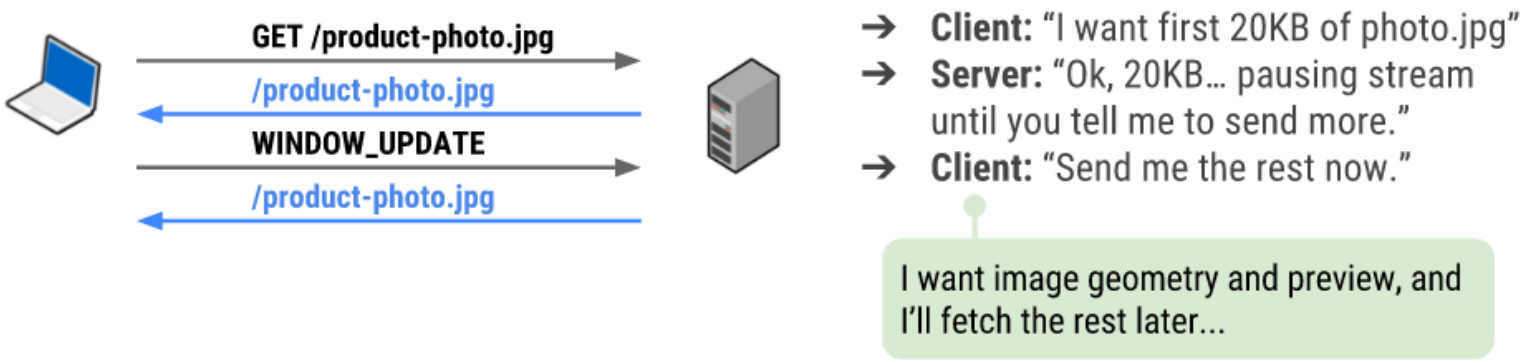
HTTP/2 HEADERS

- Se mantienen casi todos los HEADERs de HTTP/1.1.
- No se codifican más en ASCII.
- Surgen nuevos pseudo-headers que contiene información que estaba en el método y otros headers.
- Por ejemplo: `HEAD /algo HTTP/1.1` se reemplaza con http2:
  - `:method: head`
  - `:path: /algo`
  - `:scheme:` https o http
  - `:authority: www.site.com` reemplaza al header `Host:`.

  Para las respuestas: `:status:` códigos de retornos 200, 301, 404, etc.

HTTP/2 priorización y flow-control

- Los streams dentro de una misma conexión tienen flow-control individual.
- Los streams pueden tener un weight (prioridad).
- Los streams pueden estar asociados de forma jerárquica, dependencias.



fuente: https://docs.google.com/presentation/d/1r7QXGYOLCh4fcUq0jDdDwKJWNqWK1o4xMtYpKZCJYjM/present?slide=id.p19

HTTP/2 inline vs. push

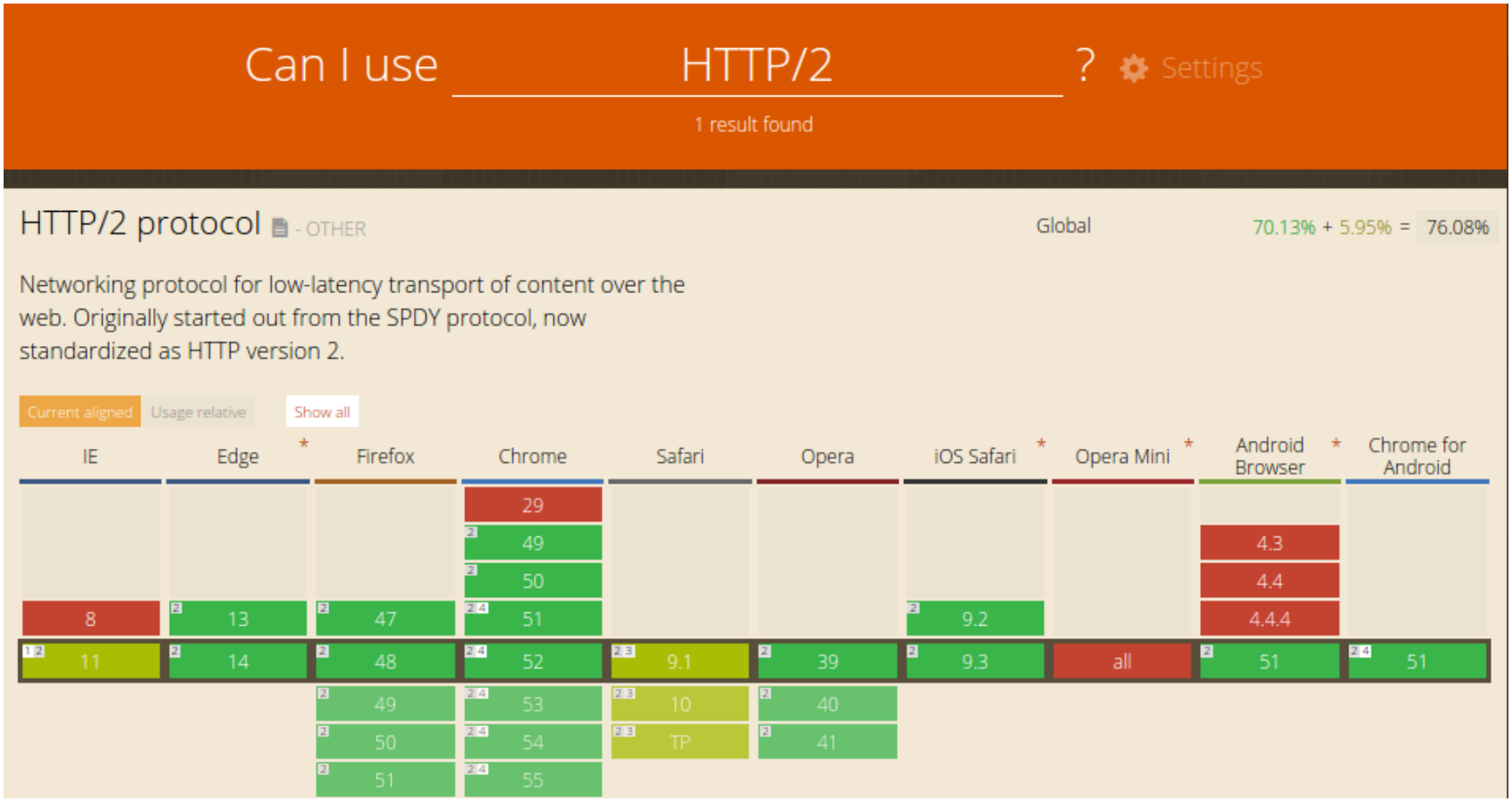
- Cuando el cliente solicita una página, “parsea” el primer response HTML luego solicita el resto.
- El server puede enviar el HTML más otros datos, por ejemplo CSS o Javascript.
- No siempre es lo que necesita el cliente, depende de que funcionalidad ofrece.

fuente: https://docs.google.com/presentation/d/1r7QXGYOLCh4fcUq0jDdDwKJWNqWK1o4xMtYpKZCJYjM/present?slide=id.p19

Compresión y Soporte

- Compresión de encabezados.
- SPDY/2 propone usar GZIP.
- GZIP + cifrado, tiene “bugs” utilizados por atacantes.
- Se crea un nuevo compresor de Headers: HPACK.
- H2 y SPDY, soportados en la mayoría de los navegadores.

Soporte en clientes para 2016



fuente: http://caniuse.com/#search=HTTP %2F2

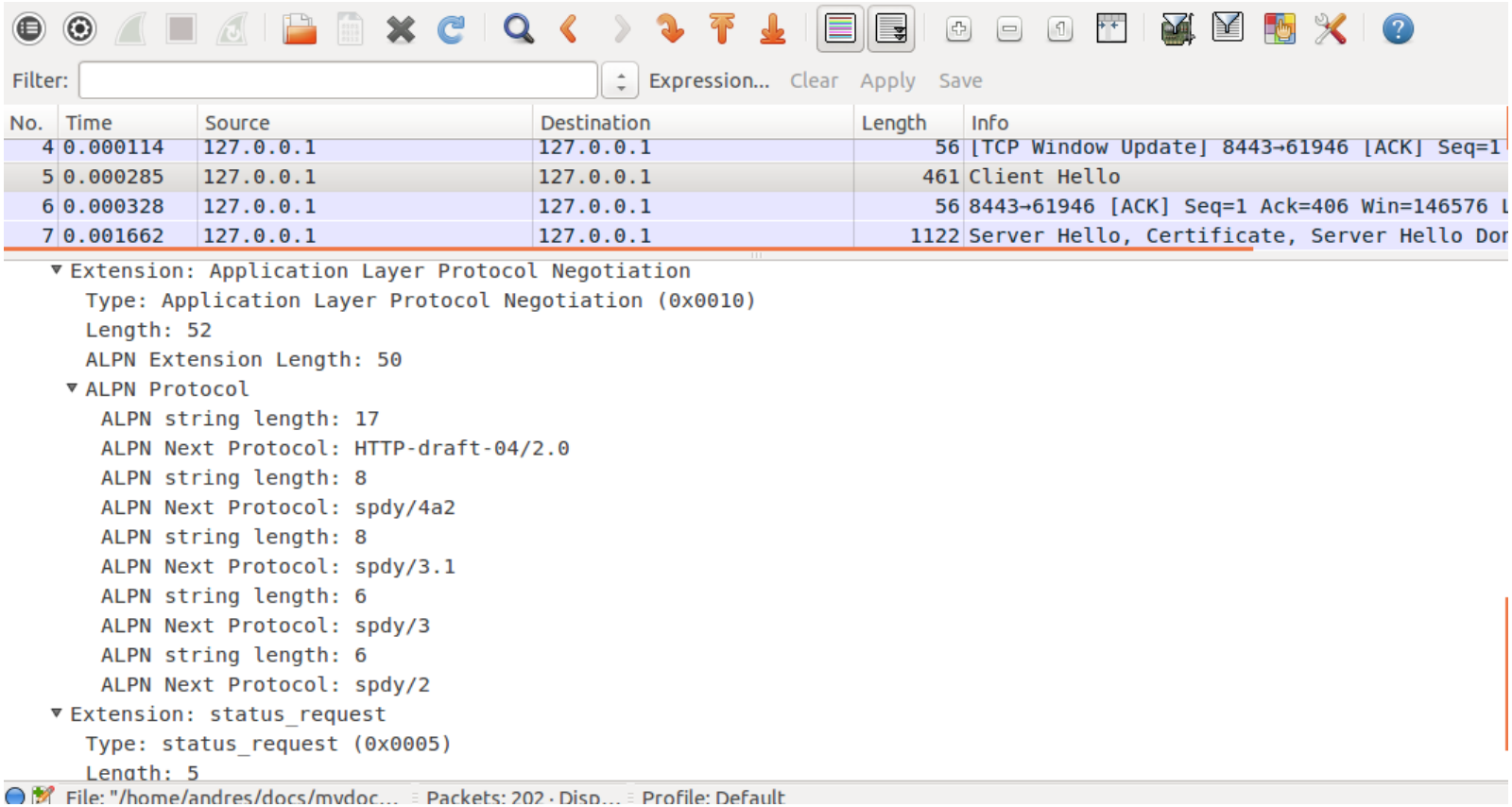
Otras Características

- HTTP/1.1, posibilidad de hacer un upgrade durante la conexión:
  Upgrade Header. `Connection: Upgrade, HTTP2-Settings`
  `Upgrade: h2c|h2`.

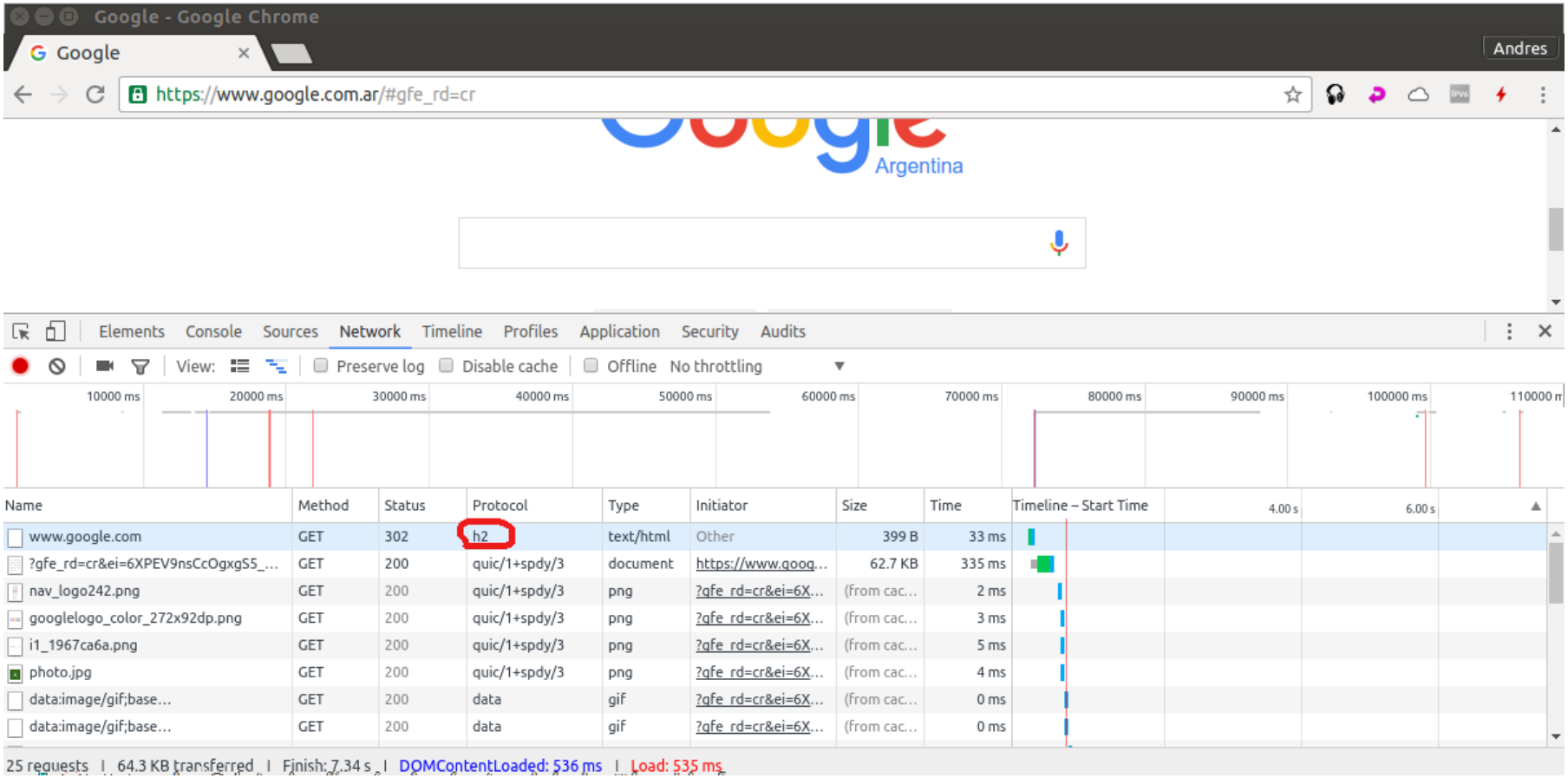
- http2: Negociar el protocolo de aplicación:
  ALPN: Application-Layer Protocol Negotiation.
  Se negocia como extensión de SSL en Hello (Anteriormente NPN). Se ofrece: h2, h3, http/1.1, ...

- Posibilidad de negociar protocolo alternativo:
  Alterantive Service: `alt-svc`.

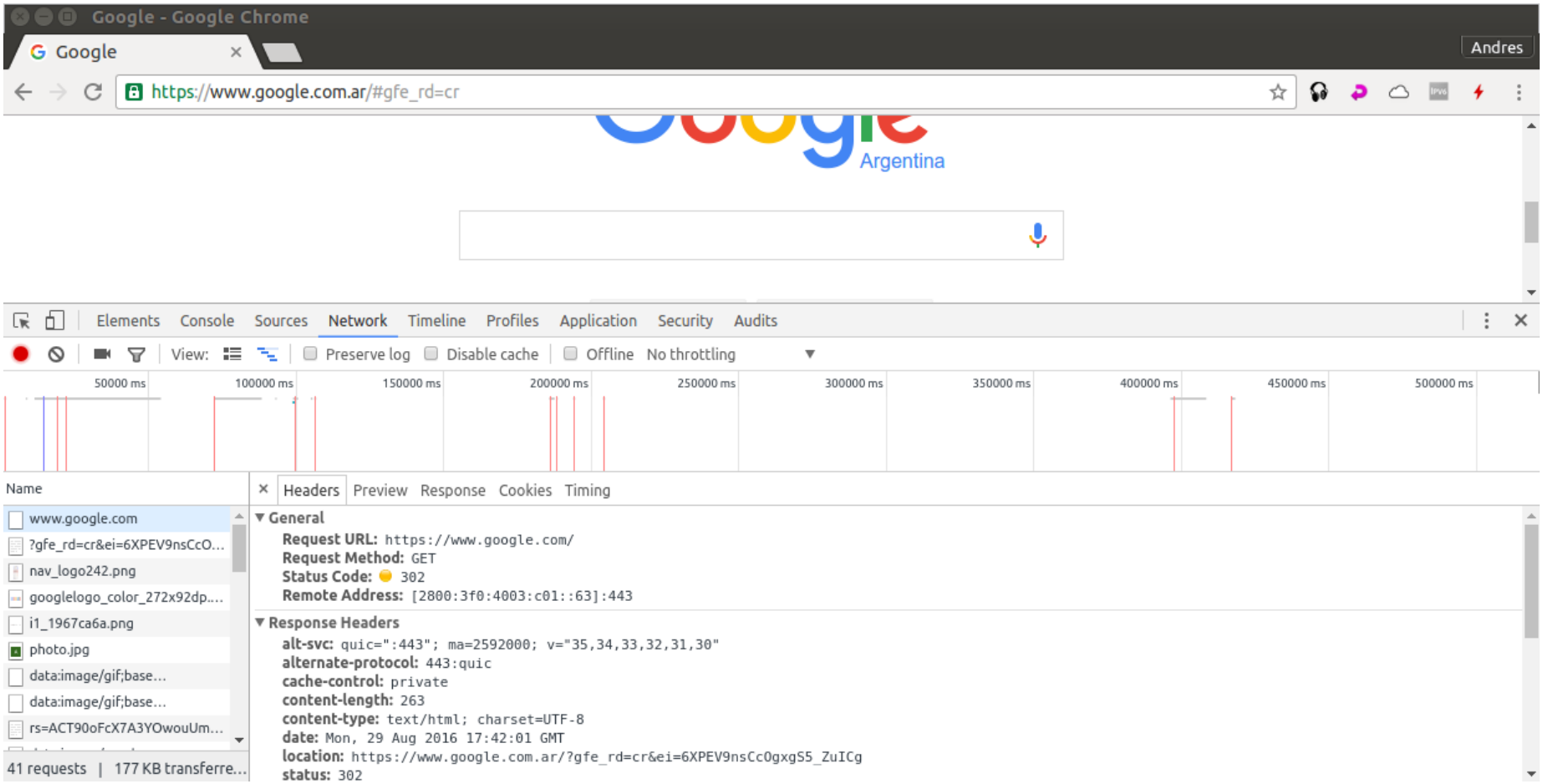
Application-Layer Protocol Neg.



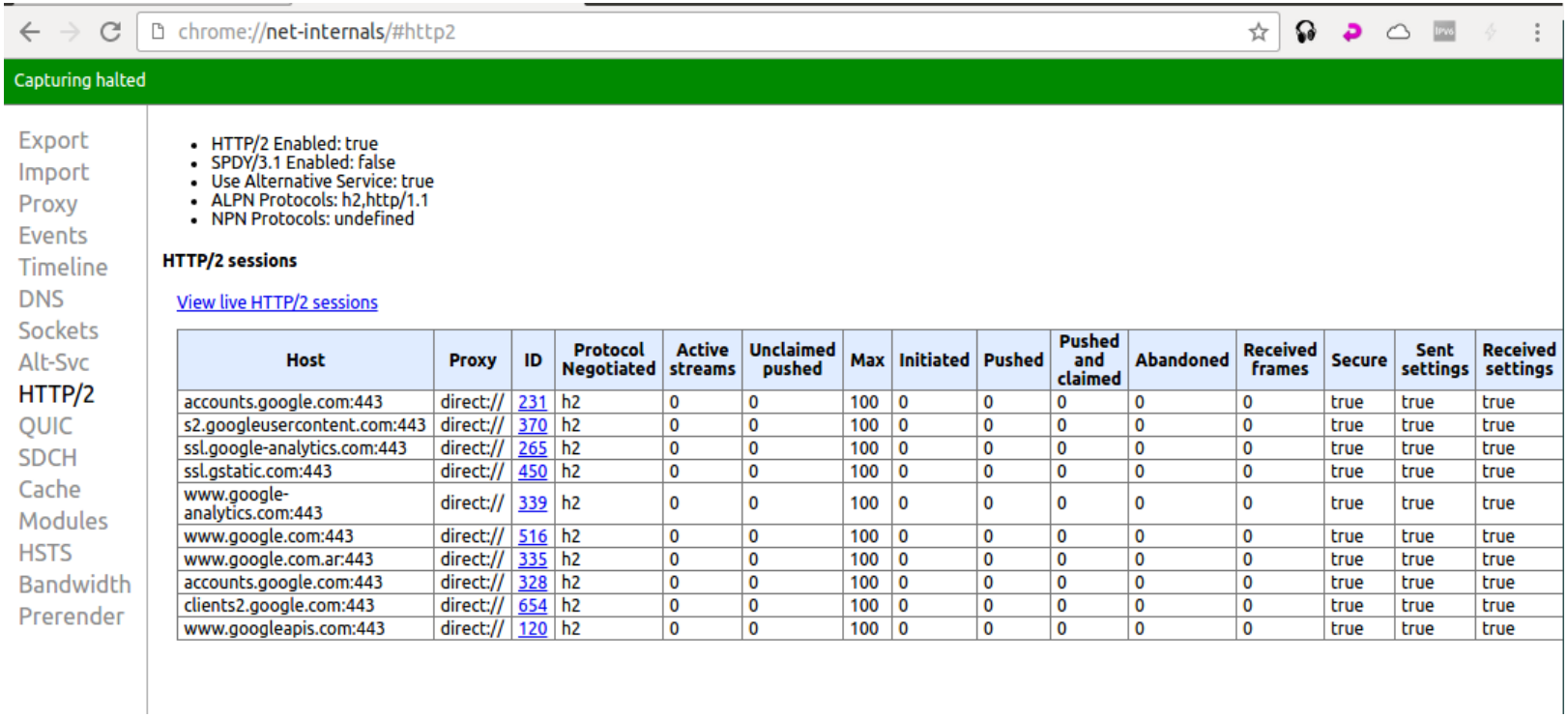
Debugging (2016)



Debugging (Cont.)

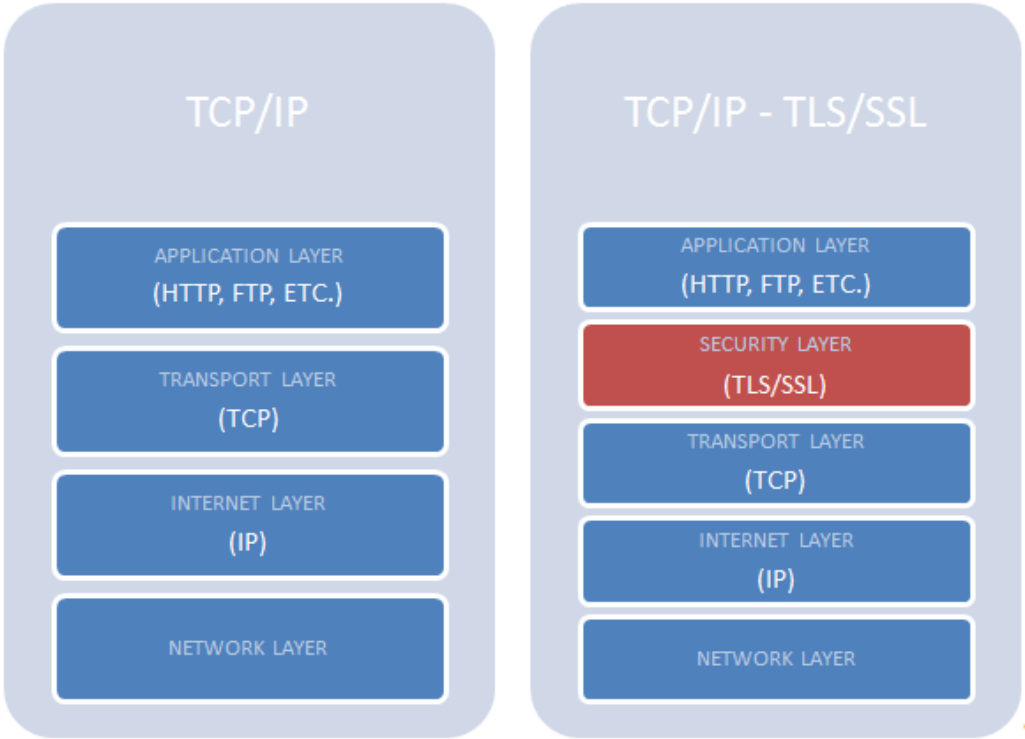


Debugging (Cont.)



Host	Proxy	ID	Protocol Negotiated	Active streams	Unclaimed pushed	Max	Initiated	Pushed	Pushed and claimed	Abandoned	Received frames	Secure	Sent settings	Received settings
accounts.google.com:443	direct://	231	h2	0	0	100	0	0	0	0	0	true	true	true
s2.googleusercontent.com:443	direct://	370	h2	0	0	100	0	0	0	0	0	true	true	true
ssl.google-analytics.com:443	direct://	265	h2	0	0	100	0	0	0	0	0	true	true	true
ssl.gstatic.com:443	direct://	450	h2	0	0	100	0	0	0	0	0	true	true	true
www.google-analytics.com:443	direct://	339	h2	0	0	100	0	0	0	0	0	true	true	true
www.google.com:443	direct://	516	h2	0	0	100	0	0	0	0	0	true	true	true
www.google.com.ar:443	direct://	335	h2	0	0	100	0	0	0	0	0	true	true	true
accounts.google.com:443	direct://	328	h2	0	0	100	0	0	0	0	0	true	true	true
clients2.google.com:443	direct://	654	h2	0	0	100	0	0	0	0	0	true	true	true
www.googleapis.com:443	direct://	120	h2	0	0	100	0	0	0	0	0	true	true	true

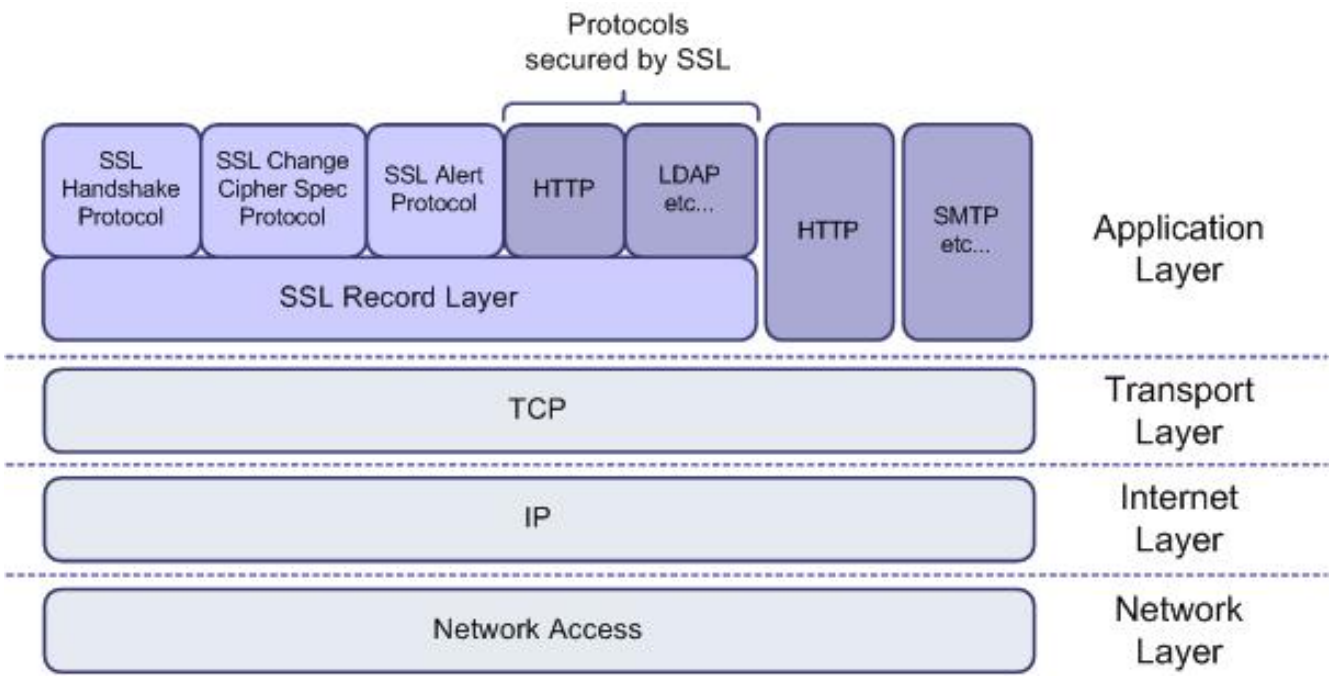
SSL/TLS



Network Layer = Network Access Layer = Network Interface Layer = Link + Física

fuente: https://www.simple-talk.com/dotnet/net-framework/tlssl-and-net-framework-4-0/

SSL/TLS (Cont.)

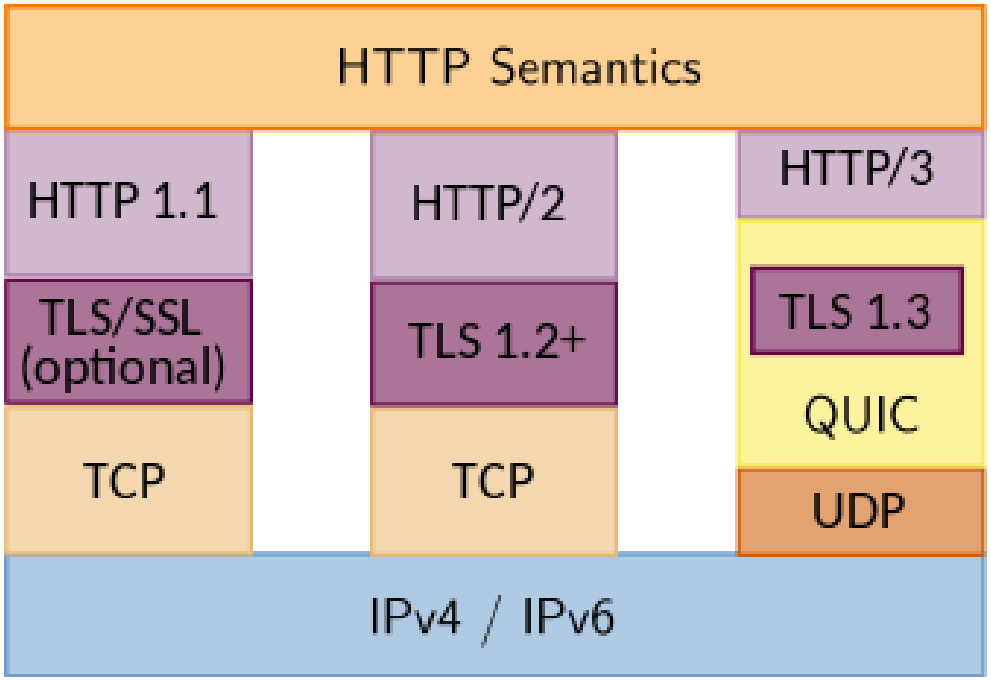


fuente: http://nicolascormier.com/documentation/bin/apache/apache2_with_ssl_tls/part1.htm

Qué es HTTP/3?

- Transportar HTTP sobre nuevo protocolo: QUIC, que corre sobre UDP, default UDP:443.
- Igual que HTTP/2 se conservan métodos y semántica de HTTP/1.1.
- Base del trabajo protocolo desarrollado por Google, QUIC.
- Definido en:
  - RFC9000: QUIC: A UDP-Based Multiplexed and Secure Transport.
  - RFC9114: HTTP/3.

HTTP/3



fuente: https://en.wikipedia.org/wiki/HTTP/3

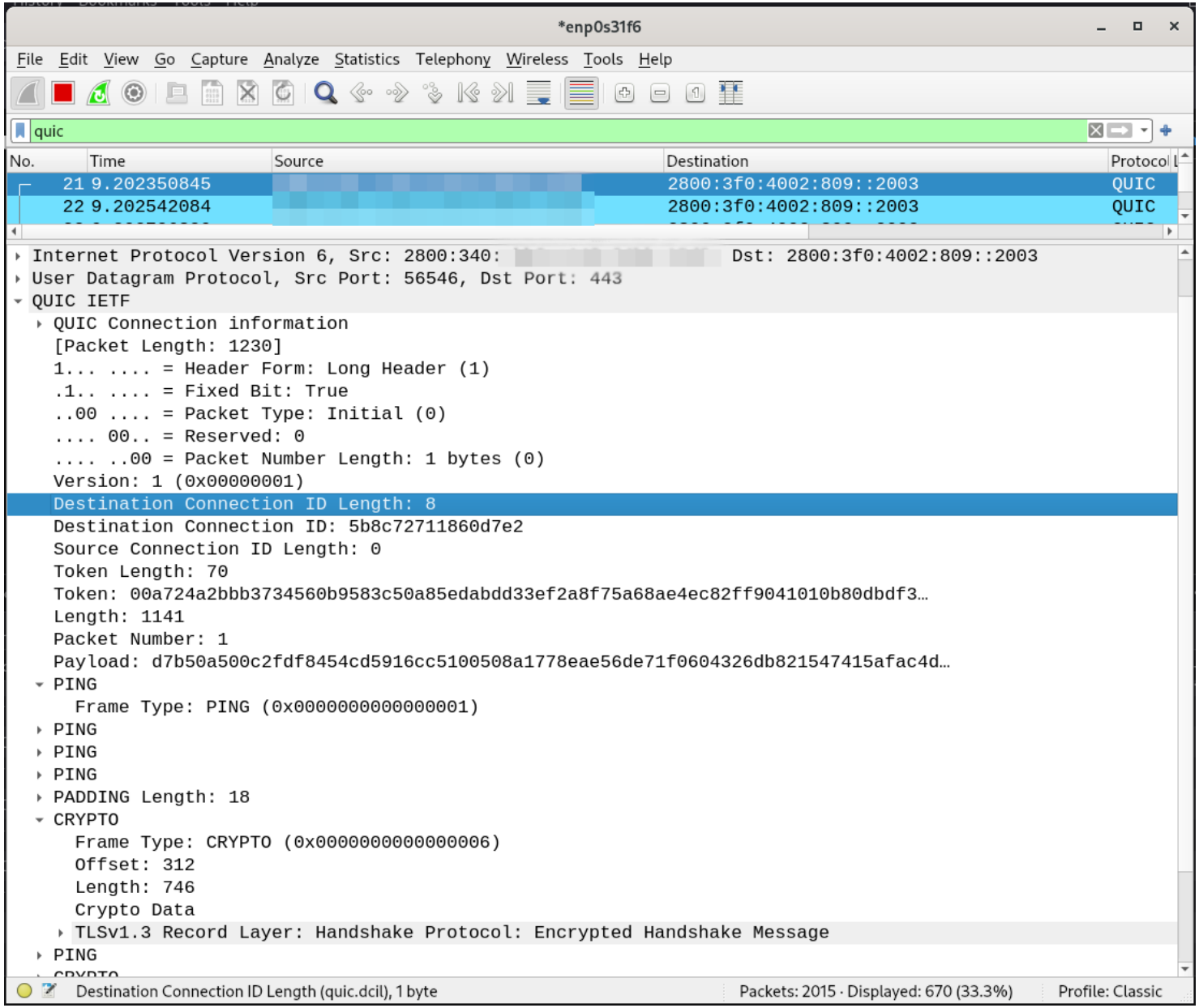
Diferencias principales de HTTP/3-QUIC

- Reducción de latencia en conexiones (2 msj. de handshake).
- No depende del manejo de la conexión y controles de TCP (evita HOLB).
- Loss recovery/congestion control, sobre UDP, lo puede hacer como TCP-CUBIC o de otra forma.
- QUIC genere nuevos números de secuencia y re-cifra las retransmisiones, permitiendo mejor detección de pérdidas y medición de RTT.
- QUIC Paquetes cifrados de forma individual, no por conexión/bytestream.
- QUIC facilita movilidad, tiene ID de conexión independiente de IPs y ports.
- Solo trabaja seguro con cifrado.

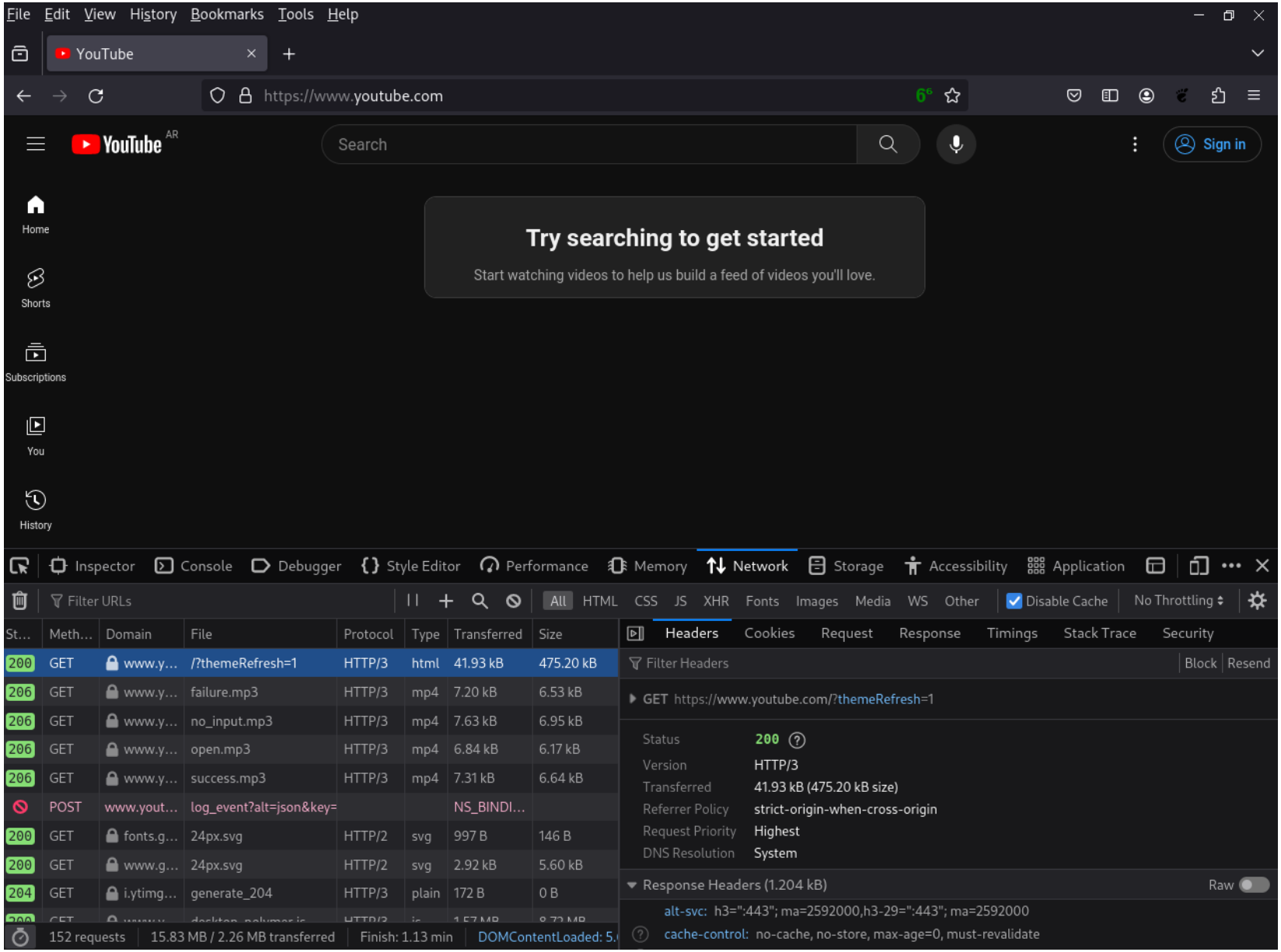
Diferencias principales de HTTP/3-QUIC (Cont.)

- Desventaja de QUIC: muchos middle-boxes filtran UDP, salvo port 53 y NAT.
- Protocolo de transporte QUIC: implementado usualmente en User-space vs Kernel-space.
- UDP facilita algunos ataques de seguridad.

HTTP/3 captura



HTTP/3 Debugging (2024)



[HTTP/2] https://http2.github.io/.

[Ilya Grigorik] HTTP/2 is here, let's optimze!
https://docs.google.com/presentation/d/1r7QXGYOLCh4fcUq0jDdDwKJWNqWK1o4xMtYpKZCJYjM/present?slide=id.p19.

[HTTP/2-dev] https://web.dev/performance-http2/#streams,-messages,-and-frames

[HTTP/2-undertow] https://undertow.io/blog/2015/04/27/An-in-overview-of-HTTP2.html

[HTTP/3-cloudflare] https://www.cloudflare.com/learning/performance/what-is-http3/

[HTTP/3-wikipedia] https://en.wikipedia.org/wiki/HTTP/3

[QUIC-wikipedia] https://en.wikipedia.org/wiki/QUIC

[QUIC-google] https://peering.google.com/#/learn-more/quic