

Redes de Computadoras 2020

Capa 5 DNS, HTTP

Alumnos:

Losano Quintana, Juan Cruz (locxhalosano45@gmail.com)

Piñero, Tomás Santiago (tom-300@hotmail.com)

Docentes:

Natasha Tomattis (natasha.tomattis@mi.unc.edu.ar)

Ayudantes alumnos:

Aguerreberry Matthew, Sulca Sergio, Moral Ramiro, Soriano Juan

Mayo, 2020

Requisitos

- Computadora por cada 2 personas

Consignas

DNS

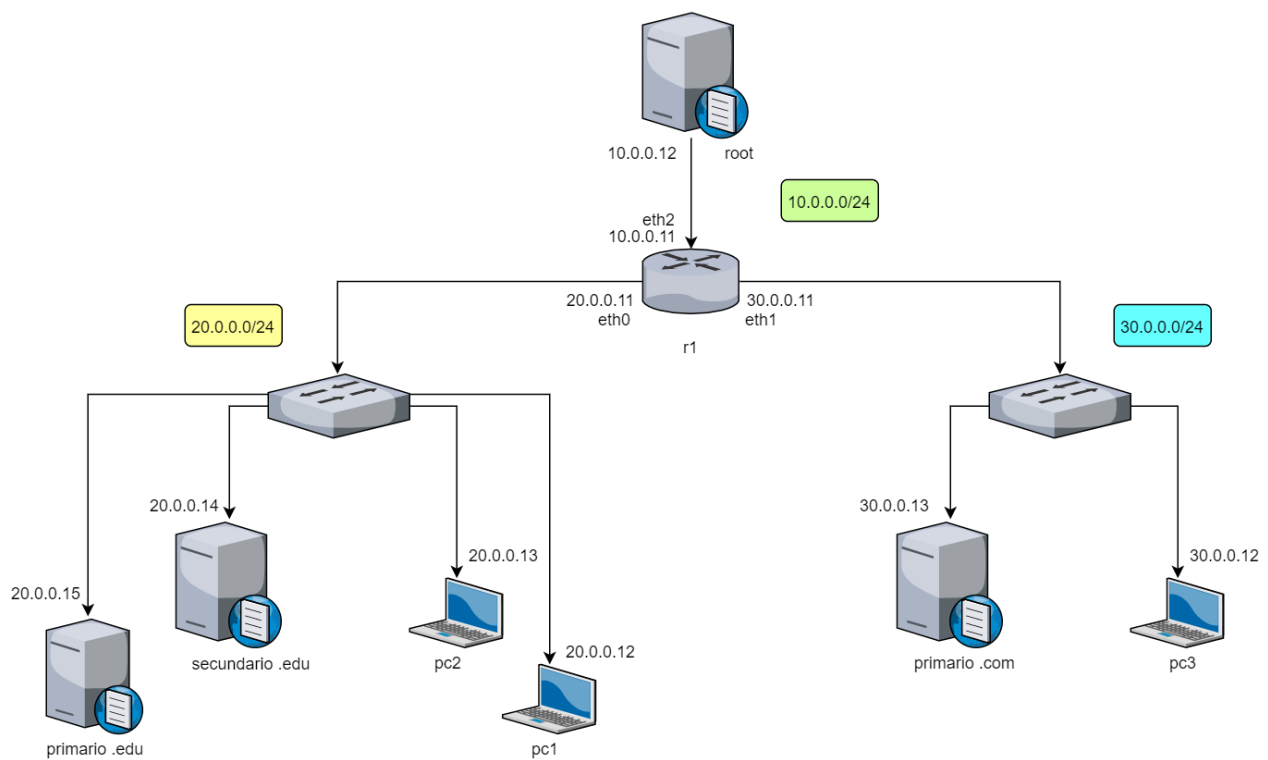


Fig. 1 - Topología.

1. Implementar la topología. Primario.edu y secundario.edu son servidores master y slave sobre el dominio .edu. Primario.com es autoritativo sobre .com.

- a. Comprobar la resolución de nombres con nslookup de pc1.edu y pc2.edu a pc3.com, y viceversa. Como es el proceso de resolución, se realizan consultas iterativas o recursivas?

```
jclq@jclq-VMware:~/Documents/fcefyfyn/redes/tp5$ docker exec -ti pc1 ash
/ # nslookup pc3.p.com
Server:          20.0.0.15
Address:         20.0.0.15:53

Non-authoritative answer:
Name:   pc3.p.com
Address: 30.0.0.12
```

Fig. 2 - Resultado del comando nslookup desde la pc1 a la pc3.

```
jclq@jclq-VMware:~/Documents/fcefyfyn/redes/tp5$ docker exec -ti pc3 ash
/ # nslookup pc2.p.edu
Server:          30.0.0.13
Address:         30.0.0.13:53

Non-authoritative answer:

Non-authoritative answer:
Name:   pc2.p.edu
Address: 20.0.0.13
```

Fig. 3 - Resultado del comando nslookup desde la pc3 a la pc2.

43	153.37136...	20.0.0.12	20.0.0.15	DNS	69	Standard query 0x9534 A pc3.p.com
44	153.37140...	20.0.0.12	20.0.0.15	DNS	69	Standard query 0x961b AAAA pc3.p.com
47	153.37188...	20.0.0.15	10.0.0.12	DNS	92	Standard query 0xeef8 A pc3.p.com OPT
48	153.37217...	20.0.0.15	10.0.0.12	DNS	92	Standard query 0x8e90 AAAA pc3.p.com OPT
49	153.37242...	10.0.0.12	20.0.0.15	DNS	142	Standard query response 0x8e90 AAAA pc3.p.com NS ns1.p.com A 30.0.0.13 OPT
50	153.37301...	20.0.0.15	10.0.0.12	DNS	108	Standard query 0xa20b AAAA pc3.p.com OPT
51	153.37309...	20.0.0.15	192.33.4.12	DNS	82	Standard query 0x3598 NS <Root> OPT
52	153.37312...	20.0.0.12	20.0.0.15	TCP	110	Time-to-live exceeded (time to live exceeded in transit)
53	153.37323...	10.0.0.12	20.0.0.15	DNS	142	Standard query response 0xeef8 A pc3.p.com NS ns1.p.com A 30.0.0.13 OPT
54	153.37326...	10.0.0.12	20.0.0.15	DNS	142	Standard query response 0xa20b AAAA pc3.p.com NS ns1.p.com A 30.0.0.13 OPT
55	153.37348...	20.0.0.15	10.0.0.12	DNS	108	Standard query 0xbdbd A pc3.p.com OPT
56	153.37363...	10.0.0.12	20.0.0.15	DNS	142	Standard query response 0xbdbd A pc3.p.com NS ns1.p.com A 30.0.0.13 OPT
57	153.37399...	20.0.0.15	30.0.0.13	DNS	92	Standard query 0x1861 AAAA pc3.p.com OPT
58	153.37422...	20.0.0.15	30.0.0.13	DNS	92	Standard query 0x7d47 A pc3.p.com OPT
59	153.37437...	30.0.0.13	20.0.0.15	DNS	124	Standard query response 0x7d47 A pc3.p.com A 30.0.0.12 OPT
60	153.37444...	30.0.0.13	20.0.0.15	DNS	167	Standard query response 0x1861 AAAA pc3.p.com SOA ns1.p.com.p.com OPT
61	153.37459...	20.0.0.15	20.0.0.12	DNS	85	Standard query response 0x9534 A pc3.p.com A 30.0.0.12

Fig. 4 - Sniffeeo de Wireshark de la consulta "nslookup".

Como se puede ver en la Fig. 4, las consultas que se realizan son recursivas, ya que el cliente le pregunta a su nameserver por el dominio y éste le consulta al root, que le responde con la zona pedida. Por último, el name server devuelve la respuesta al cliente.

- b. Por qué implementaría un servidor DNS secundario o esclavo?

Se implementa un servidor DNS secundario o esclavo por motivos de backup y para reducir la carga de los Servidores DNS principales. Si se cae el DNS primario, el secundario responde las consultas.

- c. En pc1 configurar la cache local con dnsmasq, comprobar el funcionamiento de la caché.

```
/ # dig -u +noall +stats pc3.p.com
;; Query time: 3926 usec
;; SERVER: 20.0.0.15#53(20.0.0.15)
;; WHEN: Thu May 21 20:19:11 UTC 2020
;; MSG SIZE rcvd: 82

/ # dnsmasq --listen-address=127.0.0.1
/ # dig -u +noall +stats pc3.p.com
;; Query time: 405 usec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu May 21 20:19:28 UTC 2020
;; MSG SIZE rcvd: 82

/ # nslookup pc3.p.com
Server:      127.0.0.1
Address:     127.0.0.1#53

Non-authoritative answer:
Name:   pc3.p.com
Address: 30.0.0.12
```

Fig. 5 - Cache local en pc1.

La Fig. 5 muestra la caché local de pc1 con dnsmasq funcionando. Antes de habilitar la caché el tiempo de la consulta fue de casi 4 milisegundos (3.926 exactamente) y, luego de habilitar la caché, 405 microsegundos.

- d. Para qué sirve el directorio /etc/hosts?

El directorio “/etc/hosts” sirve para especificar la resolución de nombres textuales a direcciones IP.

```
green@virma:~$ cat /etc/hosts
127.0.0.1      localhost
::1           localhost
127.0.1.1     pop-os.localdomain  pop-os
```

Fig. 6 - Resultado del comando “cat /etc/hosts” en la VM.

2. ¿Qué son las consultas recursivas e iterativas en DNS?

En una consulta recursiva, un cliente solicita a un servidor DNS que obtenga por sí mismo la respuesta completa al dominio pedido. Se anidan las consultas.

En una consulta iterativa, el servidor DNS no otorga una respuesta completa sino que devuelve las direcciones IP de los servidores de nivel superior, por lo que el cliente ahora debe realizar una nueva consulta a uno de estos servidores. Esto se hace hasta recibir la respuesta completa.

3. Analizando el tráfico DNS con Wireshark y utilizando dig +trace a cualquier dominio con ipv6 (por ejemplo youtube.com) se pide:
- a. Identificar el primer datagrama UDP donde se encuentra la consulta DNS por el dominio.

dns cname == youtube						
No.	Time	Source	Destination	Protocol	Window	Info
156	0.767544916	192.168.31.128	192.168.31.2	DNS		Standard query 0x82ff A www.youtube.com OPT
157	0.767818801	192.168.31.128	192.168.31.2	DNS		Standard query 0xc0eb AAAA www.youtube.com OPT

▶ Frame 157: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface ens33, id 0
▶ Ethernet II, Src: VMware_87:6a:5e (00:0c:29:87:6a:5e), Dst: VMware_f6:11:3b (00:50:56:f6:11:3b)
▶ Internet Protocol Version 4, Src: 192.168.31.128, Dst: 192.168.31.2
▶ User Datagram Protocol, Src Port: 51107, Dst Port: 53
▼ Domain Name System (query)
Transaction ID: 0xc0eb
Flags: 0x0100 Standard query
0... .. = Response: Message is a query
.000 0... .. = Opcode: Standard query (0)
... ..0... .. = Truncated: Message is not truncated
... ..1... .. = Recursion desired: Do query recursively
... ..0... .. = Z: reserved (0)
... ..0... .. = Non-authenticated data: Unacceptable
Questions: 1
Answer RRs: 0
Authority RRs: 0
Additional RRs: 1
▼ Queries
▶ www.youtube.com: type AAAA, class IN
Name: www.youtube.com
[Name Length: 15]
[Label Count: 3]
Type: AAAA (IPv6 Address) (28)
Class: IN (0x0001)
▶ Additional records
[Response In: 221]

Fig. 7 - Consulta IPv6 a por www.youtube.com

- b. En qué capa tcp/ip se encuentra dicha consulta? Explique la información que muestra en pantalla dig +trace.

Se encuentra en la capa 5 (aplicación).

Dig +trace muestra todas las respuestas a las consultas iterativas que se realizan para obtener el nombre pedido, en este caso www.youtube.com. El resultado del comando puede verse en la Fig. 8. se ve como se resuelve el . luego el com. Y por ultimo youtube.com.

```
jclq@jclq-VMware:~$ dig @8.8.4.4 +trace www.youtube.com AAAA

; <<>> DiG 9.11.3-ubuntu1.11-Ubuntu <<>> @8.8.4.4 +trace www.youtube.com AAAA
; (1 server found)
;; global options: +cmd
.           87174    IN      NS      a.root-servers.net.
.           87174    IN      NS      b.root-servers.net.
.           87174    IN      NS      c.root-servers.net.
.           87174    IN      NS      d.root-servers.net.
.           87174    IN      NS      e.root-servers.net.
.           87174    IN      NS      f.root-servers.net.
.           87174    IN      NS      g.root-servers.net.
.           87174    IN      NS      h.root-servers.net.
.           87174    IN      NS      i.root-servers.net.
.           87174    IN      NS      j.root-servers.net.
.           87174    IN      NS      k.root-servers.net.
.           87174    IN      NS      l.root-servers.net.
.           87174    IN      NS      m.root-servers.net.
.           87174    IN      RRSIG   NS 8 0 518400 20200530170000 20
JVvwqFzzpQWfNufRySOAtfwr4X+QNVv+y4LuXGUHkKSmLgbTPIHk0jG hBHe0ZgBrjTNYJrA7ht96Fj
;; Received 525 bytes from 8.8.4.4#53(8.8.4.4) in 34 ms

com.        172800   IN      NS      a.gtld-servers.net.
com.        172800   IN      NS      b.gtld-servers.net.
com.        172800   IN      NS      c.gtld-servers.net.
com.        172800   IN      NS      d.gtld-servers.net.
com.        172800   IN      NS      e.gtld-servers.net.
com.        172800   IN      NS      f.gtld-servers.net.
com.        172800   IN      NS      g.gtld-servers.net.
com.        172800   IN      NS      h.gtld-servers.net.
com.        172800   IN      NS      i.gtld-servers.net.
com.        172800   IN      NS      j.gtld-servers.net.
com.        172800   IN      NS      k.gtld-servers.net.
com.        172800   IN      NS      l.gtld-servers.net.
com.        172800   IN      NS      m.gtld-servers.net.
com.        86400   IN      DS      30909 8 2 E2D3C916F6DEEAC73294E
com.        86400   IN      RRSIG   DS 8 1 86400 20200530170000 202
mq8p8uez3SVUa3Gq6zX+WZNR7NZ4Uqp4eVPviPOT+7/u+MMotafYCP 8k8tjFArSKKrSZq/t43atSyj
;; Received 1175 bytes from 192.58.128.30#53(j.root-servers.net) in 31 ms

youtube.com. 172800   IN      NS      ns2.google.com.
youtube.com. 172800   IN      NS      ns1.google.com.
youtube.com. 172800   IN      NS      ns3.google.com.
youtube.com. 172800   IN      NS      ns4.google.com.
CK0P0JMG874LJREF7EFN8430QVIT8BSM.com. 86400 IN NSEC3 1 1 0 - CK0Q1GIN43N1ARRC90
CK0P0JMG874LJREF7EFN8430QVIT8BSM.com. 86400 IN RRSIG NSEC3 8 2 86400 2020052104
y3WkOBn9 huGPmwnMwiD6ueYSXrIMRX+dvCqV76um4wFw/gYIsil/jw==
HSAIN06JFIFDFKM2P9HTBFN9U0GA6UCV.com. 86400 IN NSEC3 1 1 0 - H5AK5VB6Q09VQA8LIA
HSAIN06JFIFDFKM2P9HTBFN9U0GA6UCV.com. 86400 IN RRSIG NSEC3 8 2 86400 2020052406
G0857LCQ 6X3jsH1nuVAzjIDiE22UPaEzvQgNPYn+xmrB4A1+dwL2lg==
;; Received 848 bytes from 192.41.162.30#53(l.gtld-servers.net) in 157 ms

www.youtube.com. 86400   IN      CNAME   youtube-ui.l.google.com.
youtube-ui.l.google.com. 300   IN      AAAA    2800:3f0:4002:80a::200e
;; Received 106 bytes from 216.239.38.10#53(ns4.google.com) in 175 ms
```

Fig. 8 - Resultado de “dig @8.8.4.4 +trace www.youtube.com AAAA”.

- c. Identificar el datagrama UDP donde finalmente se encuentre la respuesta a dicha consulta. ¿Qué tipo de registro almacena dicha respuesta?

```
16 1.036436340 192.168.31.2 192.168.31.128 DNS Standard query response 0x1c8e AAAA www.youtube.com CNAME you...
Frame 16: 148 bytes on wire (1184 bits), 148 bytes captured (1184 bits) on interface ens33, id 0
Ethernet II, Src: VMware_f6:11:3b (00:50:56:f6:11:3b), Dst: VMware_87:6a:5e (00:0c:29:87:6a:5e)
Internet Protocol Version 4, Src: 192.168.31.2, Dst: 192.168.31.128
User Datagram Protocol, Src Port: 53, Dst Port: 39588
Domain Name System (response)
Transaction ID: 0x1c8e
Flags: 0x8180 Standard query response, No error
1... .. = Response: Message is a response
.000 0... .. = Opcode: Standard query (0)
... .. = Authoritative: Server is not an authority for domain
... .. = Truncated: Message is not truncated
... ..1... .. = Recursion desired: Do query recursively
... ..1... .. = Recursion available: Server can do recursive queries
... ..0... .. = Z: reserved (0)
... ..0... .. = Answer authenticated: Answer/authority portion was not authenticated by the server
... ..0... .. = Non-authenticated data: Unacceptable
... ..0000 = Reply code: No error (0)
Questions: 1
Answer RRs: 2
Authority RRs: 0
Additional RRs: 1
Queries
  www.youtube.com: type AAAA, class IN
    Name: www.youtube.com
    [Name Length: 15]
    [Label Count: 3]
    Type: AAAA (IPv6 Address) (28)
    Class: IN (0x0001)
Answers
  www.youtube.com: type CNAME, class IN, cname youtube-ui-1.google.com
  youtube-ui-1.google.com: type AAAA, class IN, addr 2800:3f0:4002:80a::200e
    Name: youtube-ui-1.google.com
    Type: AAAA (IPv6 Address) (28)
    Class: IN (0x0001)
    Time to live: 5 (5 seconds)
    Data length: 16
    AAAA Address: 2800:3f0:4002:80a::200e
Additional records
  <Root>: type OPT
  [Request In: 4]
  [Time: 0.003644528 seconds]
```

Fig. 9 - Respuesta de DNS.

La respuesta almacena dos tipos de registros: CNAME, que es el nombre que se pidió buscar, y AAAA, que contiene la dirección IPv6 de dicho nombre.

4. ¿A qué denominamos zona de autoridad en DNS? ¿Qué almacena un registro NS y cuál es su importancia?

La zona de autoridad es una parte de espacios de nombres de dominios en la que un servidor DNS es responsable, es decir, donde el servidor tiene autoridad sobre la zona del espacio de nombres dado. Por ejemplo la univesidad con dominio purdue.edu es autoritativa sobre la zona purdue.edu como asi tambien cs.purdue.edu, que es la página del departamento de ciencias de la computación de purdue.

El registro de servidor de nombres (NS) determina los servidores que comunicarán la información del DNS de un dominio. Es importante porque indica al servidor DNS si es responsable de una solicitud o no, es decir, si tiene que organizar la zona en cuestión, o bien a quién tiene que reenviar la solicitud.

5. Qué puede decir sobre seguridad en DNS. (DNSSEC)

Se debe configurar correctamente la comunicación entre servidor DNS primario y secundario, para evitar solicitudes de transferencia de zona que no provengan de servidores específicos indicados por el administrador.

DNSSEC (Domain Name System Security Extensions) tiene como objetivo asegurar al usuario una conexión a un sitio real u a otro servicio que corresponda a un nombre de dominio en particular.

HTTP

1. En la topología anterior transformar al 'pc2' en un web server.
 - a. Dicho web server debe correr una aplicación web desarrollada con Flask (un framework escrito en python).

```
from flask import Flask
from flask import request
import json as j

app = Flask(__name__)

lista_alumnos = {"12345678" : "Juan Manuel de Losano",
                  "87654321" : "Lucas White"}

@app.route('/')
def root():
    return 'Hola, Nati!'

@app.route('/alumnos', methods=['GET', 'POST'])
def alumnos():
    if request.method == 'GET':
        json = j.dumps(lista_alumnos)
        return json
    else:
        ID = request.form['id']
        name = request.form['nombre']

        lista_alumnos[ID] = name
        return 'AGREGA2\n'

@app.route('/alumnos/<id_a>', methods=['GET', 'DELETE'])
def id(id_a):
    result = lista_alumnos.get(id_a, 0)
    if result == 0:
        return 'No existe el alumno'
    elif request.method == 'GET':
        return lista_alumnos[id_a]
    else: #Es un DELETE
        lista_alumnos.pop(id_a)
        return 'ELIMINA2\n'
```

Fig. 10 - Flask API.

- b. La aplicación debe administrar una lista de alumnos (No usar una base de datos, solo una lista dentro del código python). Debe brindar 2 endpoints.
 - i. “/alumnos” Para obtener la lista completa de alumnos. (GET)

```
jclq@jclq-VMware:~/Documents/fcefyf/rees/tp5$ docker exec -ti pc3 ash
/ # curl http://pc2.p.edu:5000/alumnos
{"12345678": "Juan Manuel de Rosas", "87654321": "Manuel Belgrano", "39621464": "San Martin"}/ #
```

Fig. 11 - Respuesta a GET request.

- ii. “/alumnos” Para agregar un alumno a la lista. (POST)

```
/ # curl -d "id=20202020&nombre=armando armando" -X POST http://pc2.p.edu:5000/alumnos
AGREGA2
/ # curl http://pc2.p.edu:5000/alumnos
{"12345678": "Juan Manuel de Rosas", "87654321": "Manuel Belgrano", "39621464": "San Martin", "20202020": "armando armando"}/ #
```

Fig. 12 - Resultado a POST request.

- iii. “/alumnos/id” Para eliminar un alumno con id de la lista. (DEL) Devolver el error correspondiente si el usuario no existe.

```
/ # curl -X DELETE http://pc2.p.edu:5000/alumnos/20202020
ELIMINA2
/ # curl http://pc2.p.edu:5000/alumnos
{"12345678": "Juan Manuel de Rosas", "87654321": "Manuel Belgrano", "39621464": "San Martin"}
```

Fig. 13 - Resultado a DELETE request.

- c. Analizar paquetes con wireshark.
 - i. Agregar header a la request.

```
/ # tcpdump -A -s 0 'tcp port 5000 and (((ip[2:2] - ((ip[0]&0xf)<<2)) - ((tcp[12]&0xf0)>>2)) != 0)'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
20:56:12.935043 IP 30.0.0.12.50524 > 4f7c8f7e35f1.5000: Flags [P.], seq 2619162615:2619162724,
ack 1829123702, win 502, options [nop,nop,TS val 2821419632 ecr 3294769037], length 109
E....@.?.h.....\....?.m.6v....2.....
.+rp.b/.GET /alumnos HTTP/1.1
Host: pc2.p.edu:5000
User-Agent: curl/7.67.0
Accept: */*
X-MyHeader: hola q ace
```

Fig. 14 - Captura de request con header “X-MyHeader”.

- ii. Que versión de HTTP estamos usando, como puedo cambiar la versión de HTTP?

Estamos usando HTTP 1.1, para cambiar la versión de HTTP, se utiliza la opción "--httpX", donde X es la versión que se quiere utilizar ([versiones disponibles](#)). La Fig. 15 muestra el header con HTTP2.

```
20:56:12.936260 IP 4f7c8f7e35f1.5000 > 30.0.0.12.50524: Flags [P.], seq 155:279, ack 109, win 509, options [nop,nop,TS val 3294769038 ecr 2821419633], length 124
E....@.C.....\m.7...@d....2.....
.b/..+rq{"12345678": "Juan Manuel de Rosas", "87654321": "Manuel Belgrano", "39621464": "San Martin", "20202020": "armando armando"}
20:59:35.171762 IP 30.0.0.12.51136 > 4f7c8f7e35f1.5000: Flags [P.], seq 863994068:863994246, ack 1674771698, win 502, options [nop,nop,TS val 2821621869 ecr 3294971273], length 178
E...l.@.?.....3...C.....2.....
...m.eE.GET /alumnos HTTP/1.1
Host: pc2.p.edu:5000
User-Agent: curl/7.67.0
Accept: */*
Connection: Upgrade, HTTP2-Settings
Upgrade: h2c
HTTP2-Settings: AAMAAABkAARAAAAAIAAAAA
```

Fig. 15 - curl con --http2.

2. Describir 5 response code más importantes de HTTP.

- Respuestas informativas: corresponden a los números 1XX e indican una respuesta provisional. HTTP/1.0 no definió ningún estado 1XX, por lo que los servidores no deberían enviar una respuesta 1XX a un cliente HTTP/1.0 salvo para experimentos.
- Peticiones correctas: corresponden a los números 2XX, que indican que la acción solicitada por el cliente ha sido recibida, entendida, aceptada y procesada correctamente.
- Redirecciones: corresponden a los números 3XX e indican al cliente que debe tomar una acción adicional para completar el request.
- Errores del cliente: corresponden a los números 4XX, que indican que la request contiene sintaxis incorrecta o no puede procesarse.
- Errores de servidor: corresponden a los números 5XX e indican que el servidor falló al completar una request válida.

3. Una de las características de HTTP indica que es un protocolo *stateless*. Pero TCP por su contrario es un protocolo *stateful*. ¿Cómo es la relación entre ambos?

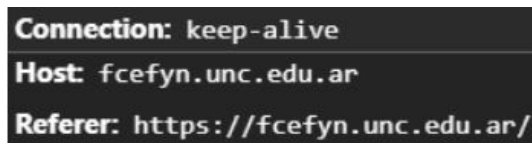
TCP es un protocolo stateful porque mantiene estados de la conexión, provee distintos beneficios: reenviar paquetes perdidos, asegurar el orden de llegada, control de flujo y congestión. Además, provee el transporte protocolos como HTTP, que es un protocolo stateless, es decir, no asegura estado de la conexión, no desea la información anterior a un request.

HTTP corre sobre TCP por lo tanto los request-response de HTTP que sean muy grandes van a hacer uso de los beneficios del protocolo TCP, así HTTP no implementa esto.

4. Explicar funcionamiento básico de las cookies. ¿Las cookies HTTP/1.1 influyen en el comportamiento del estado HTTP?

Una cookie es un dato que se manda desde una web y se almacena en el navegador web del host mientras el usuario navega en la página http. Fueron diseñadas para que las páginas web recuerden información del tipo stateful y para además seguir la actividad del usuario. Las cookies permiten que se recuerde la información de estado ya que el protocolo HTTP es stateless, por lo que sí influyen en el comportamiento de HTTP.

5. Con developer tools en Chrome (F12 o ctrl+Shift+C)
- Hacer una consulta a un sitio <http://www.portal.efn.uncor.edu/>
 - Explicar los campos Connection, Host, Referer del Request Header.



```
Connection: keep-alive
Host: fcefyn.unc.edu.ar
Referer: https://fcefyn.unc.edu.ar/
```

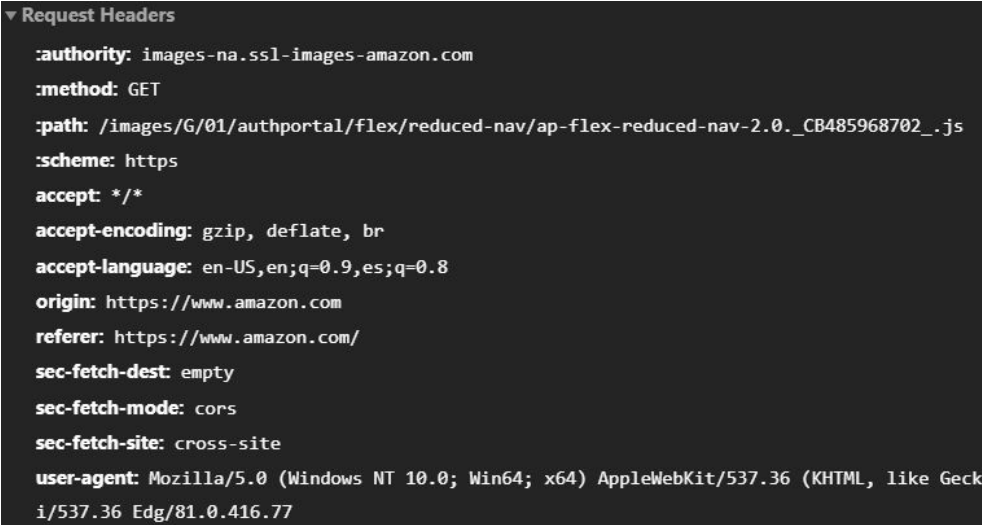
Fig. 16 - Campos Connections, host y referer del request header.

Connection: Indica si la conexión se corta o no una vez terminada la transacción HTTP. Si el valor es “keep-alive”, la conexión no se cierra, permitiendo el envío de más requests.

Host: Especifica el host y número de puerto del servidor al que se envía la request. Si el número de puerto no está especificado se da por supuesto que el puerto es el 80 (HTTP). En este caso es el servidor de la Facultad.

Referer: El Referer contiene la dirección de la página web previa desde donde se accedió al recurso. Así se puede saber desde dónde acceden los usuarios y mantener información estadística.

- b. Hacer una consulta a un sitio <https://www.amazon.com/>
 - i. Explicar los campos del Request Header.



```
▼ Request Headers
:authority: images-na.ssl-images-amazon.com
:method: GET
:path: /images/G/01/authportal/flex/reduced-nav/ap-flex-reduced-nav-2.0._CB485968702_.js
:scheme: https
accept: */*
accept-encoding: gzip, deflate, br
accept-language: en-US,en;q=0.9,es;q=0.8
origin: https://www.amazon.com
referer: https://www.amazon.com/
sec-fetch-dest: empty
sec-fetch-mode: cors
sec-fetch-site: cross-site
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.416.77
```

Fig. 17 - Campos request header de amazon.com

Los campos que empiezan y terminan con “:” se llaman campos pseudo-header y fueron implementados en HTTP/2. Los demás son HTTP/1.1

Method: indica el método utilizado por la URI.

Path: directorio donde se encuentra el archivo.

Scheme: indica el esquema con el que está formada la URI.

Accept: tipos de media que son aceptados para una respuesta.

Accept-encoding: Lista de codificaciones aceptadas.

Accept-language: Lista de idiomas aceptados para brindar una respuesta.

User-agent: es el programa que se utilizó para realizar la request. En este caso Mozilla Firefox.

- ii. Explicar los tiempos de la barra Timing.

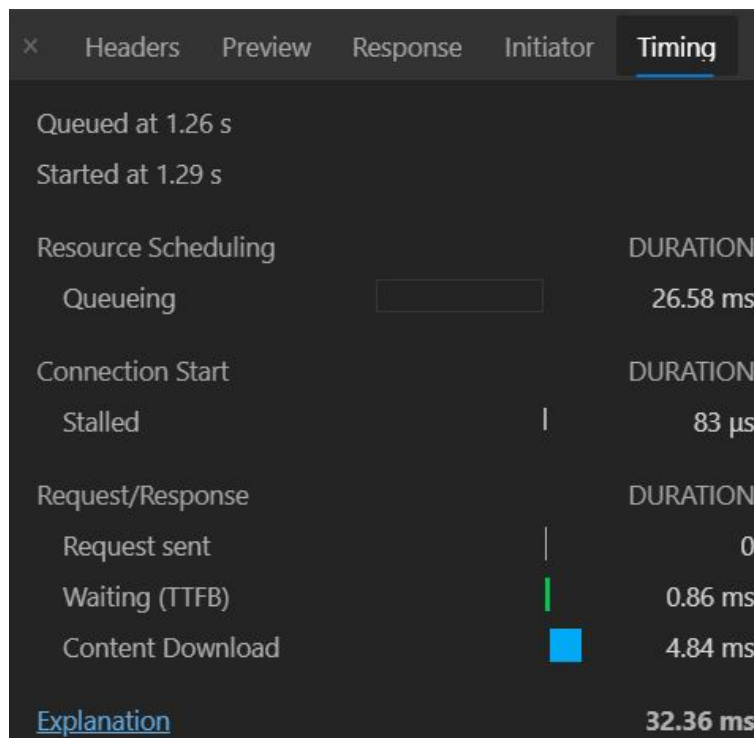


Fig. 18 - Tiempos de carga de un recurso de la página.

Queueing. Tiempo que demoró el navegador en encolar la request.

Stalled. Tiempo en que la request estuvo en la cola.

Request sent. Envía la request.

Waiting (TTFB, Time To First Byte). Tiempo de espera hasta recibir el primer byte.

Content Download. Tiempo que el navegador demoró en recibir la respuesta.

6. Mejoras de HTTP/2 con respecto a HTTP/1.1.Cuál es la diferencia de HTTP/3 con respecto a HTTPS?

HTTP/2 resuelve problemas no anticipados por los creadores de HTTP/1.1. HTTP/2 es más rápido y eficiente que HTTP/1.1, durante el proceso de carga de una web HTTP/2 establece prioridad sobre el contenido, esto permite que se cargue antes cierto contenido de la página.

La diferencia está en cómo se realiza el manejo de la seguridad. En HTTPS se utiliza TLS como encriptación de datos sobre TCP, mientras que HTTP/3 implemente un nuevo protocolo basado en UDP que integra TLS por defecto, llamado QUIC. La siguiente imagen muestra la diferencia entre las capas,

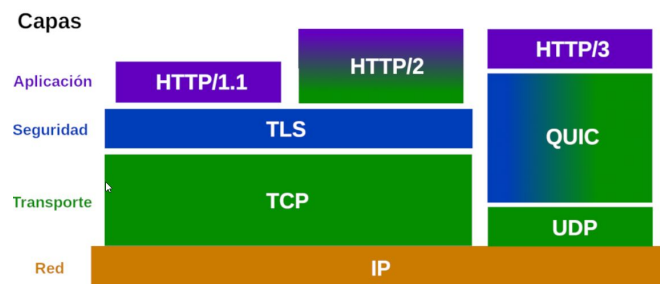


Fig. 19 - http/3 sobre QUIC.

7. Que es gRPC, cual es la diferencia con REST? Hay cambios a nivel de performance?

gRPC es un sistema de llamada a procedimiento remoto (RPC) de código abierto desarrollado inicialmente en Google. La diferencia con REST es que este último utiliza HTTP/1.1, mientras que gRPC utiliza HTTP/2.

Sí, gRPC es tiene más performance que REST porque, si bien se encontraron formas de realizar consultas concurrentemente en HTTP/1.1, HTTP/2 soporta multiplexación de consultas, permitiendo responder asíncronamente consultas concurrentes.