

Handout de Capa de Aplicación

Primer Cuatrimestre 2025

1 Primer ejercicio

El protocolo HTTP permite hacer distintos tipos de pedidos para recursos de un determinado dominio. Escriba los Requests HTTP 1.1 que permitan obtener los siguientes pedidos al sitio web del departamento de computación:

1. El recurso /
2. Encabezado del recurso /tdc
3. El recurso /logo.jpg si no fue modificado desde una determinada fecha.
4. Crear un nuevo post en el foro del departamento

1.1 Resolución

La idea es que trabajen algunas consultas simples para los ejercicios que vienen después.

1. GET / HTTP/1.1
Host: www.dc.uba.ar
2. HEAD /tdc HTTP/1.1
Host: www.dc.uba.ar
3. POST /foro HTTP/1.1
Host: www.dc.uba.ar
Content-Type: application/json
Content-Length: 70
<datos>

2 Segundo ejercicio

Suponga la siguiente página escrita en HTML que reside en el servidor www.fcen.uba.ar:

```
<html>
  <head>
    <title>Facultad de Ciencias Exactas y Naturales</title>
    <link rel="stylesheet" type="text/css" href="style.css" />
```

```

</head>
<body>
  <div>
    
    <a href="avsearch.php">  </a>
  </div>
  <div>
    <form name="searchform" action="search">
      <label>Buscar</label>
      <input name="SearchableText" type="text" title="Buscar en el Sitio" />
      <input type="image" src="search_icon.gif" />
    </form>
  </div>
</body>
</html>

```

1. ¿Cuánto tiempo en términos de RTTs transcurrirá como mínimo, hasta transferir la totalidad de la información en HTTP/1.0?
2. ¿Y en HTTP/1.1?

Resolución

1) Como es 1.0 entonces va a haber una conexión nueva para cada objeto que se deba traer del servidor y tomamos cada una como un RTT. Entonces tenemos

- 2 Rtt por el envío del html propiamente
- 2 Rtt para traer la hoja de estilos CSS
- 2 Rtt para traer searchline.png
- 2 Rtt para home.png
- 2 Rtt para searchicon.gif

avsearch.php no se trae inicialmente, se procesa su pedido cuando el cliente clickee sobre el link.

2)

En el caso de utilizar pipelining

- 1 Rtt para inicializar la conexión
- 1 Rtt para traer el HTML
- 1 Rtt para traer todas las imágenes + css

Esto sucede así porque como no se cierra la conexión para cada pedido, se podría asumir que al usar pipelining traer los archivos incluidos en el html se hace en un Rtt

En caso de no utilizar pipelining

- 1 Rtt para inicializar la conexión
- 1 Rtt para traer el HTML
- 1 Rtt por cada imágenes + css

3 Ejercicio de parcial

Desde una PC se envía un correo a `tdc-doc@dc.uba.ar` y luego se accede al recurso web `http://www.exactas.uba.ar/index.html`. La PC se conecta a la web a través de un servidor proxy con dirección IP `208.190.1.22`.

- a. Enumere los mensajes SMTP que son necesarios para el envío de un mail a `pedro@exactas.uba.ar` desde un host en el dominio `untref.edu.ar` hasta que llega al servidor de destino.
- b. Detalle los mensajes HTTP (*Requests* y *Responses*) desde que la PC envía el primer GET hasta que le llega el respectivo *Response*.

3.1 Resolución

3.1.1 Primer punto

Recordemos rápidamente cómo funciona el sistema de envío de mail.

- El cliente envía el mail a su servidor SMTP.
- Es este último el que se comunica con el SMTP destino enviándole el mail.
- El mail ya llegó a destino. El hecho de que el receptor "baje los mails" es un tema aparte, asociado a IMAP o POP
- Host → SMTP origen → SMTP destino

Con esa idea entonces veamos el orden de mensajes.

- Primero se envía el correo al servidor de correo saliente para lo que se envía la siguiente secuencia de mensajes:
 1. El cliente envía "HELO" y recibe del servidor saliente "250 Hello".
 2. El cliente envía "MAIL FROM: <usuario@untref.edu.ar>" y recibe del servidor saliente "250 Ok".
 3. El cliente envía "RCPT TO: <pedro@exactas.uba.ar>" y recibe del servidor saliente "250 Ok".
 4. El cliente envía "DATA" y recibe del servidor saliente "354 End data with <CR><LF>.<CR><LF>".
 5. El cliente envía el mail en cuestión y finaliza la transferencia con <CR><LF>.<CR><LF> y recibe del servidor saliente "250 Ok".
 6. El cliente envía "Bye" y el servidor entrante responde "221 Bye"
- Luego, el SMTP saliente envía el correo al servidor de correo entrante del dominio de destino.
 1. El servidor saliente envía "HELO" y recibe del servidor entrante "250 Hello".
 2. El servidor saliente envía "MAIL FROM: <usuario@untref.edu.ar>" y recibe del servidor entrante "250 Ok".

3. El servidor saliente envía “RCPT TO: <pedro@exactas.uba.ar>” y recibe del servidor entrante “250 Ok”.
4. El servidor saliente envía “DATA” y recibe del servidor entrante “354 End data with <CR><LF>.<CR><LF>”.
5. El servidor saliente envía el mail en cuestión y finaliza la transferencia con <CR><LF>.<CR><LF> y recibe del servidor entrante “250 Ok”.
6. El servidor saliente envía “Bye” y el servidor saliente responde “221 Bye”.

3.1.2 Segundo punto

Primero definimos el mensaje que se arma para el Request HTTP

GET /index.html HTTP/1.1
Host: www.exactas.uba.ar

Éste mensaje se envía desde el navegador Web en la PC al servidor proxy. Si tiene en su cache el recurso solicitado responde al usuario con los datos. En caso de no tenerlo en el cache, el servidor proxy establece una conexión TCP con el servidor web con nombre `www.exactas.uba.ar` y este último responde con

Response HTTP/1.1 200 OK
data

Donde **data** es el contenido del recurso en sí. Luego, el response llega al proxy quien se lo reenvía a la PC.