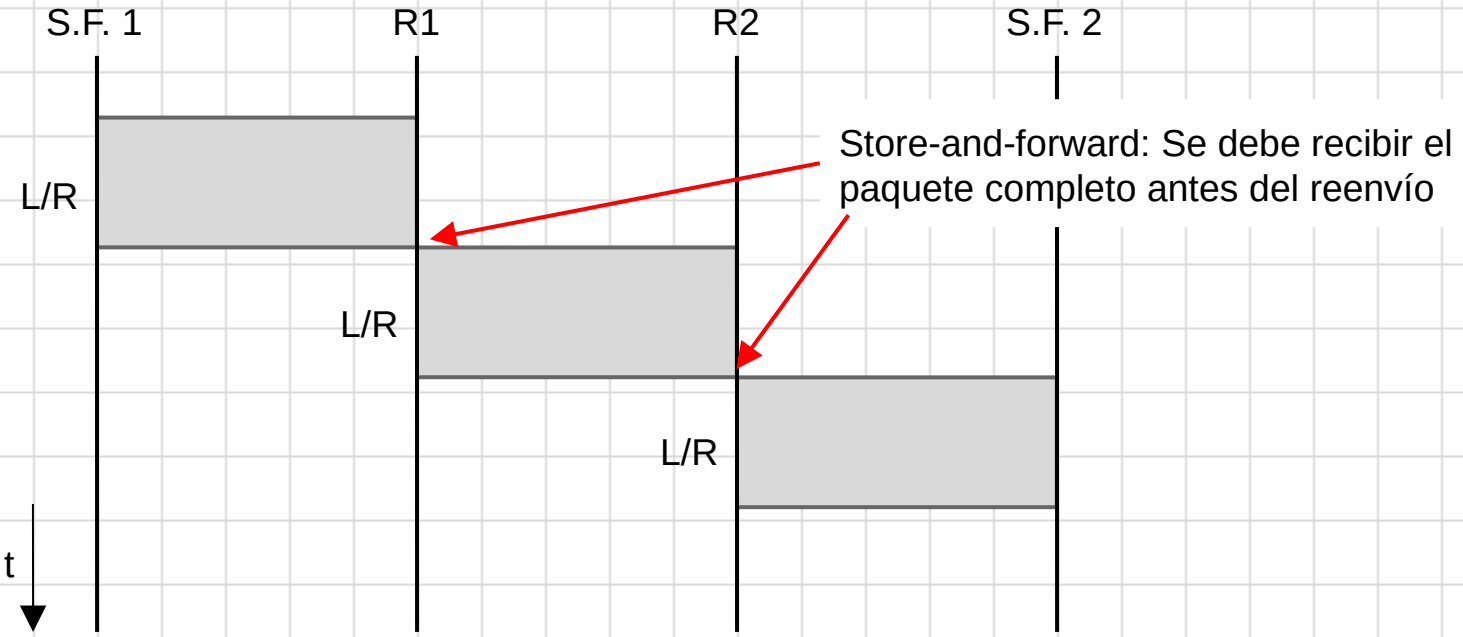
Cuaderno de Problemas

Problemas
Tema 1

Store-and-forward – Transparencia 34



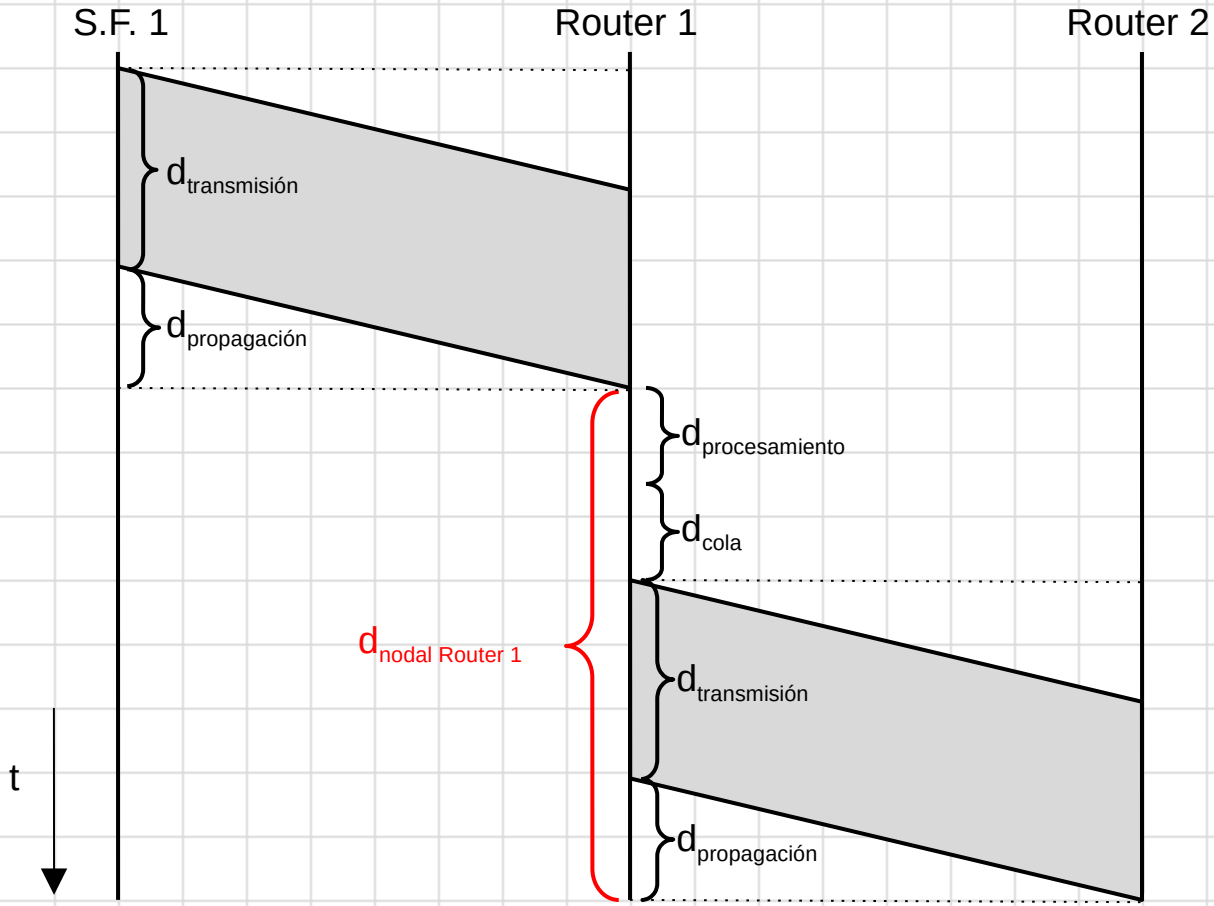
Tiempo que se tarda en transmitir un paquete de L bits sobre un enlace con ancho de banda R (bps):

$$d_{trans} = \frac{L}{R} = \frac{6Mbits}{2Mbps} = \frac{6 \times 10^6 bits}{2 \times 10^6 bps} = 3s$$

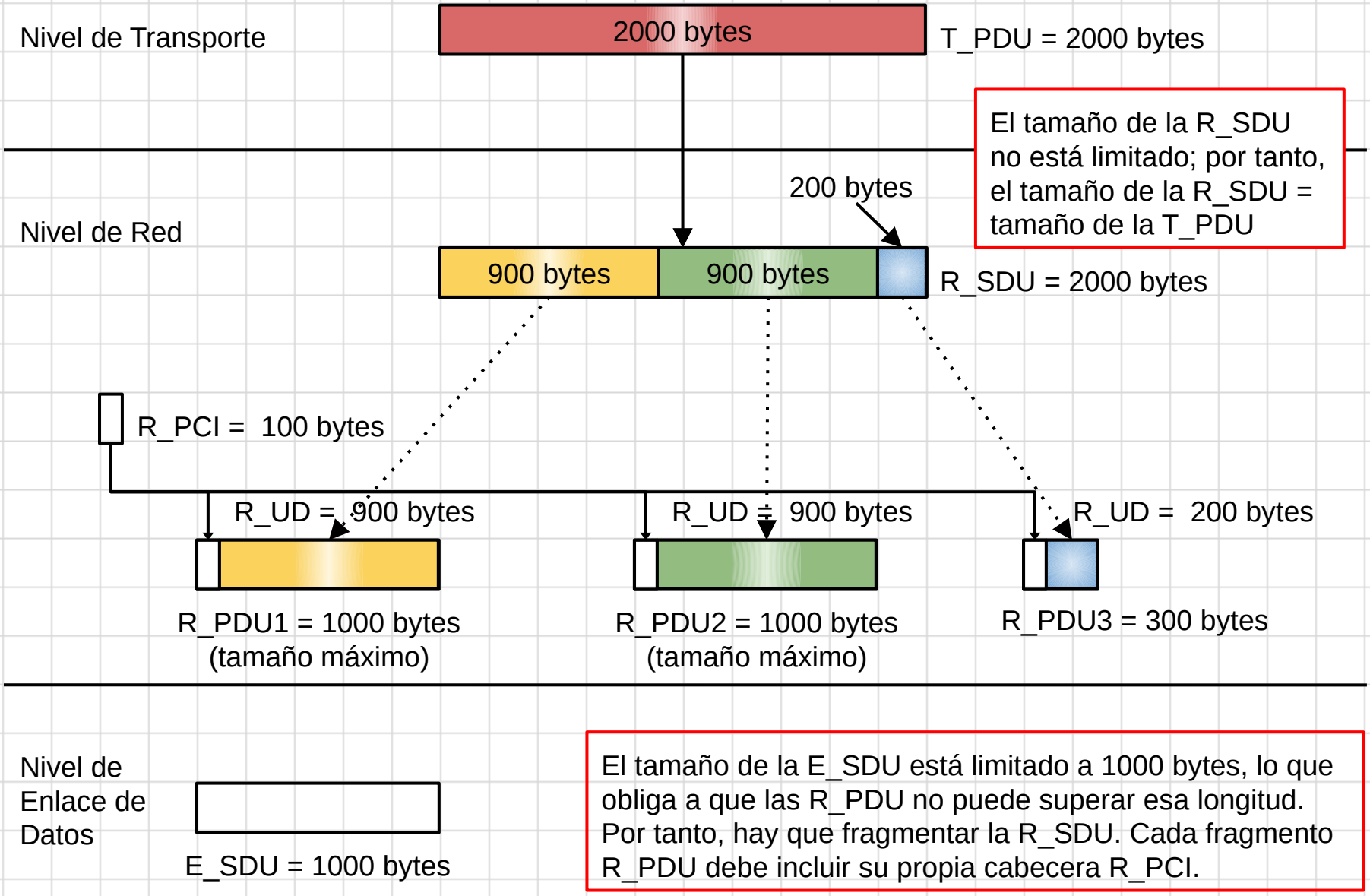
Tiempo total de extremo a extremo (de S.F. 1 a S.F. 2):

$$d_{total} = 3 \times d_{trans} = 3 \times \frac{L}{R} = 3 \times 3s = 9s$$

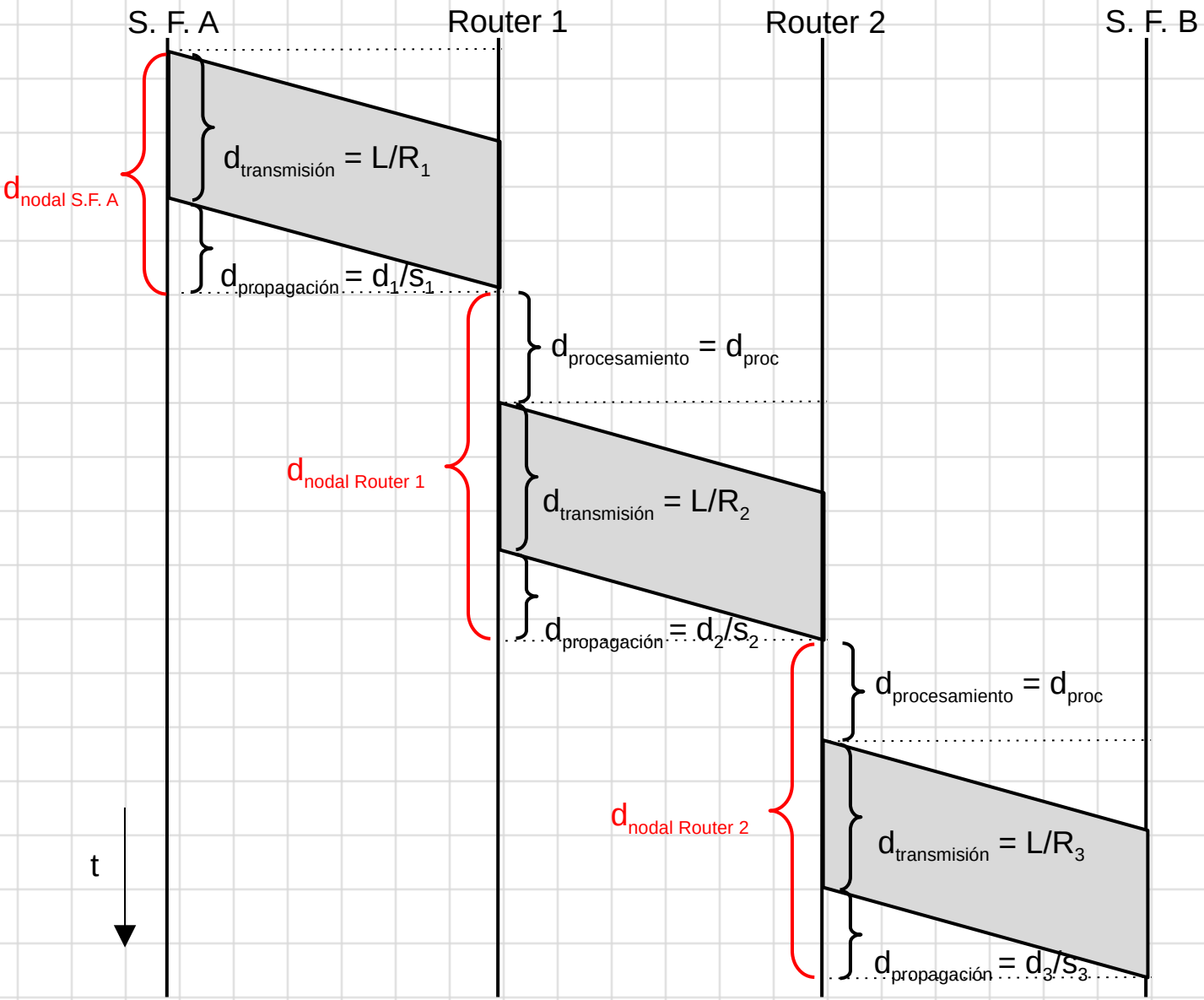
Retardos - Transparencias 41 y 42



Boletín Tema 1: P1



Boletín Tema 1: P4



De acuerdo al enunciado se asume que $d_{cola} = 0$ s

Boletín Tema 1 - P4

A partir del gráfico anterior:

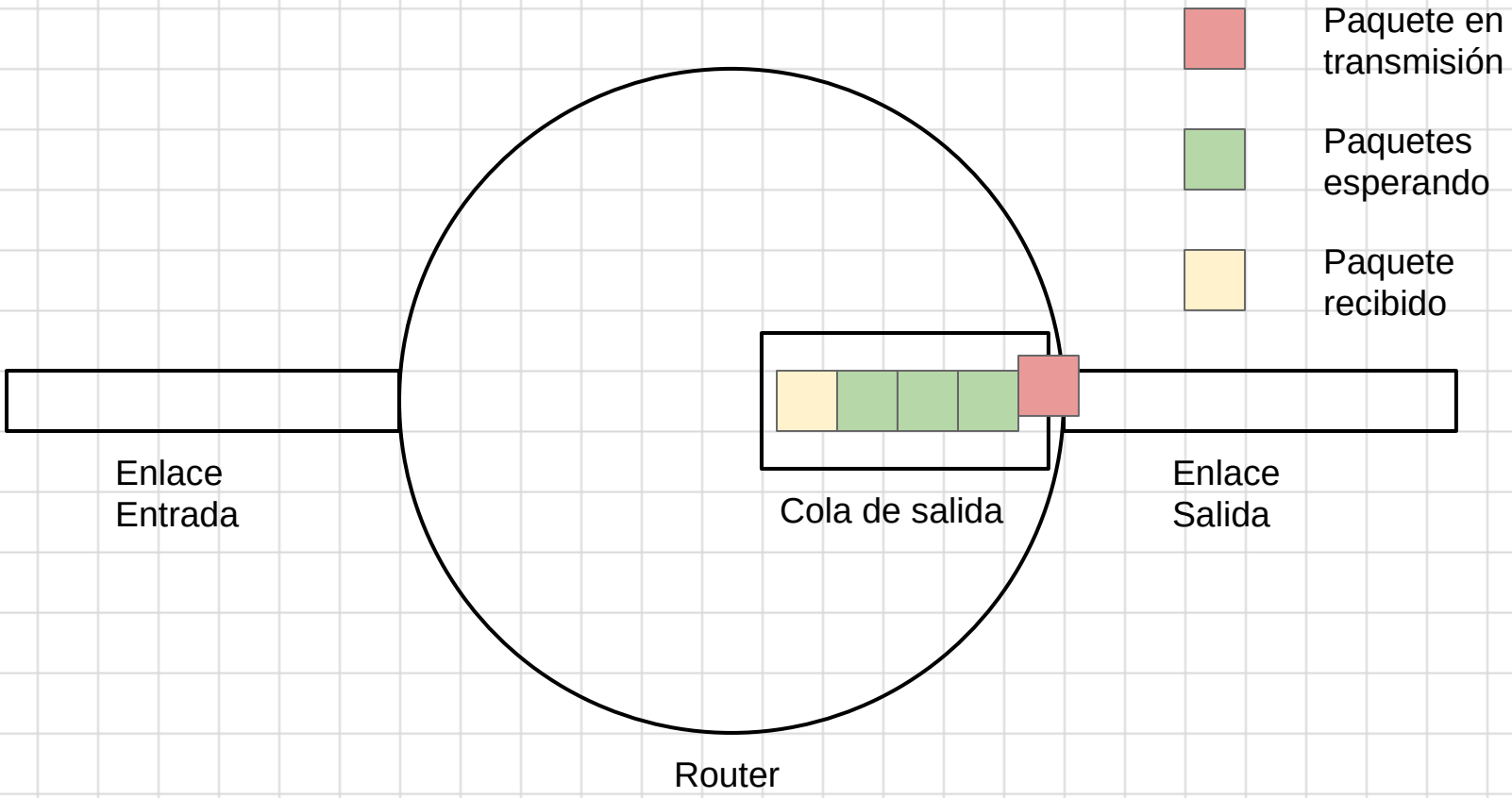
$$d_{total\ extremo\ a\ extremo} = \frac{L}{R_1} + \frac{d_1}{s_1} + d_{proc} + \frac{L}{R_2} + \frac{d_2}{s_2} + d_{proc} + \frac{L}{R_3} + \frac{d_3}{s_3}$$

Para los valores proporcionados:
L = 1500 bytes = 12000 bits
d_{proc} = 3 ms = 3 x 10^{-3} s
$s_1 = s_2 = s_3$ = 2,5 x 10^8 m/s
$R_1 = R_2 = R_3$ = 2 Mbps = 2 x 10^6 bps
d_1= 5000 km
d_2= 4000 km
d_3= 1000 km

$d_{total\ extremo\ a\ extremo}$ = 6 ms + 20 ms + 3 ms + 6 ms + 16 ms + 3 ms + 6 ms + 4 ms = 64 ms

(Nota: El alumno debe realizar los diferentes cálculos de los retardos)

Boletín Tema 1: P5



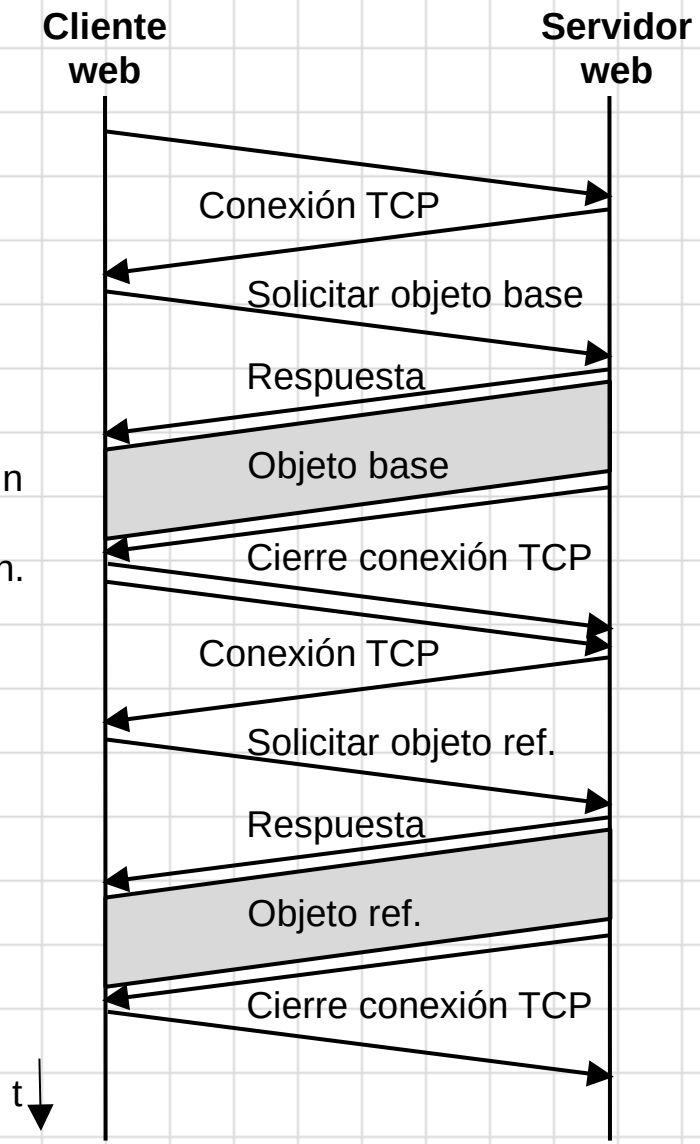
$$d_{cola} = \frac{1500\ bytes \times 3 + 750\ bytes}{2\ Mbps} = 21\ ms$$

$$d_{cola} = \frac{n \times L + (L - x)}{R}$$

Problemas
Tema 2

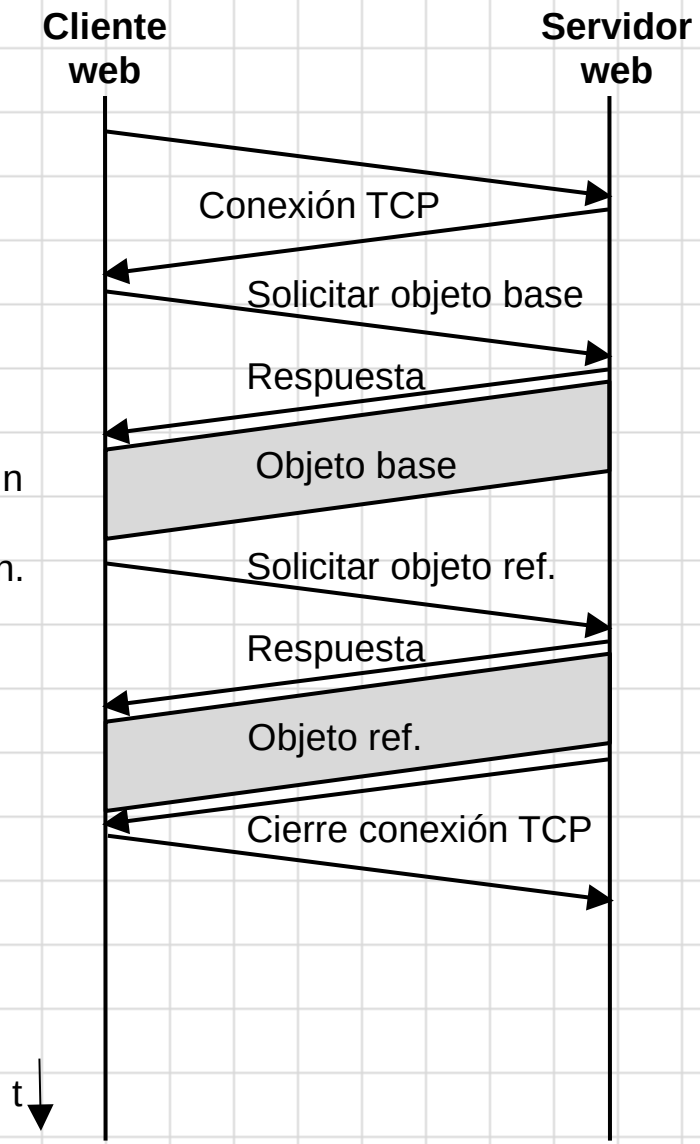
HTTP no persistente - Transparencia 35

Si el objeto base tiene un objeto referenciado hay que descargarlo también.

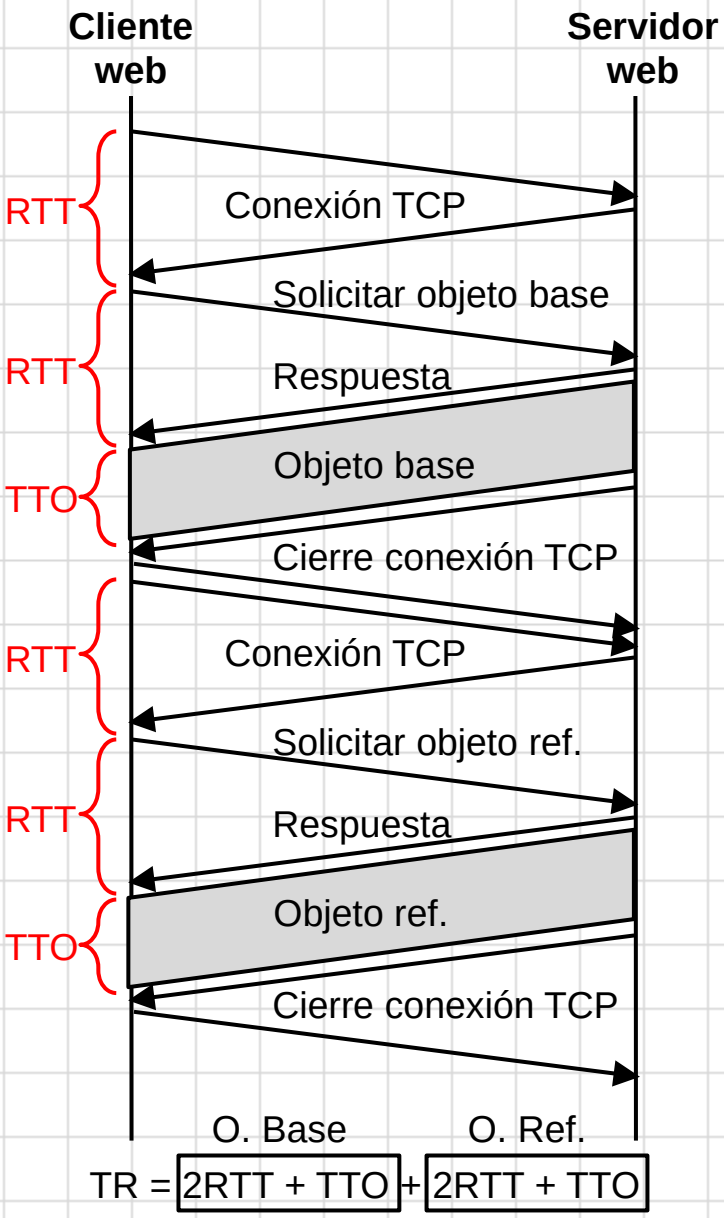


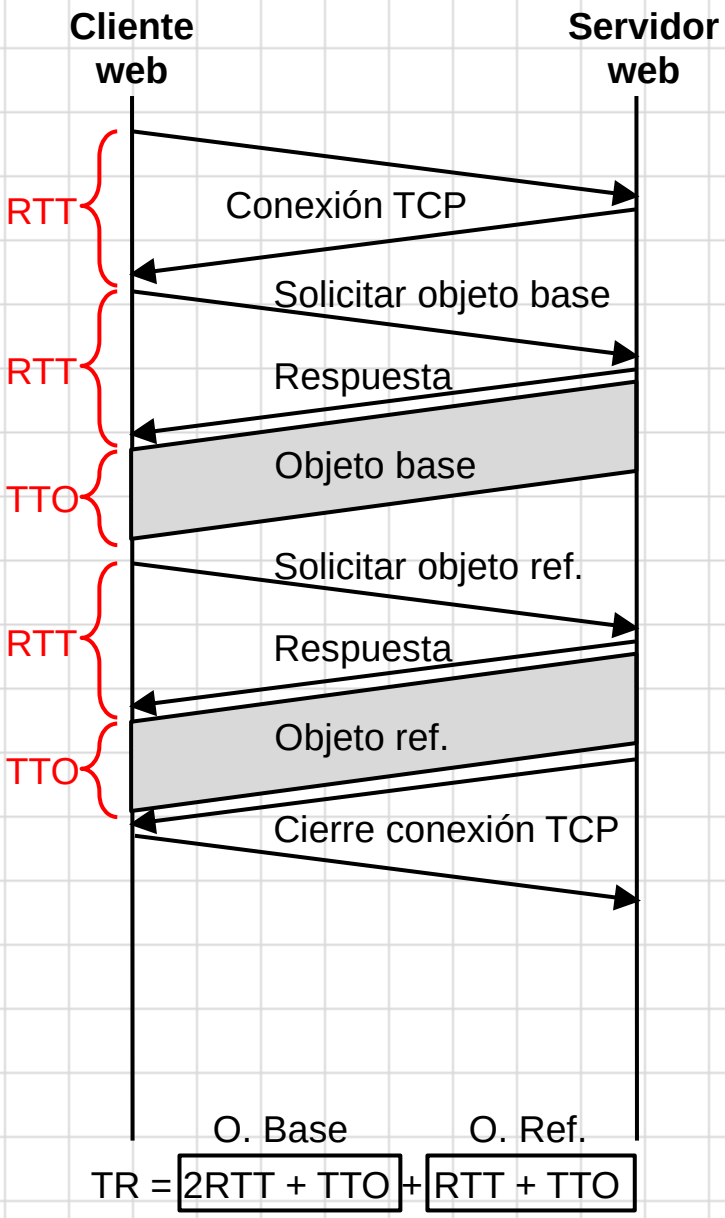
HTTP persistente - Transparencia 35

Si el objeto base tiene un objeto referenciado hay que descargarlo también.

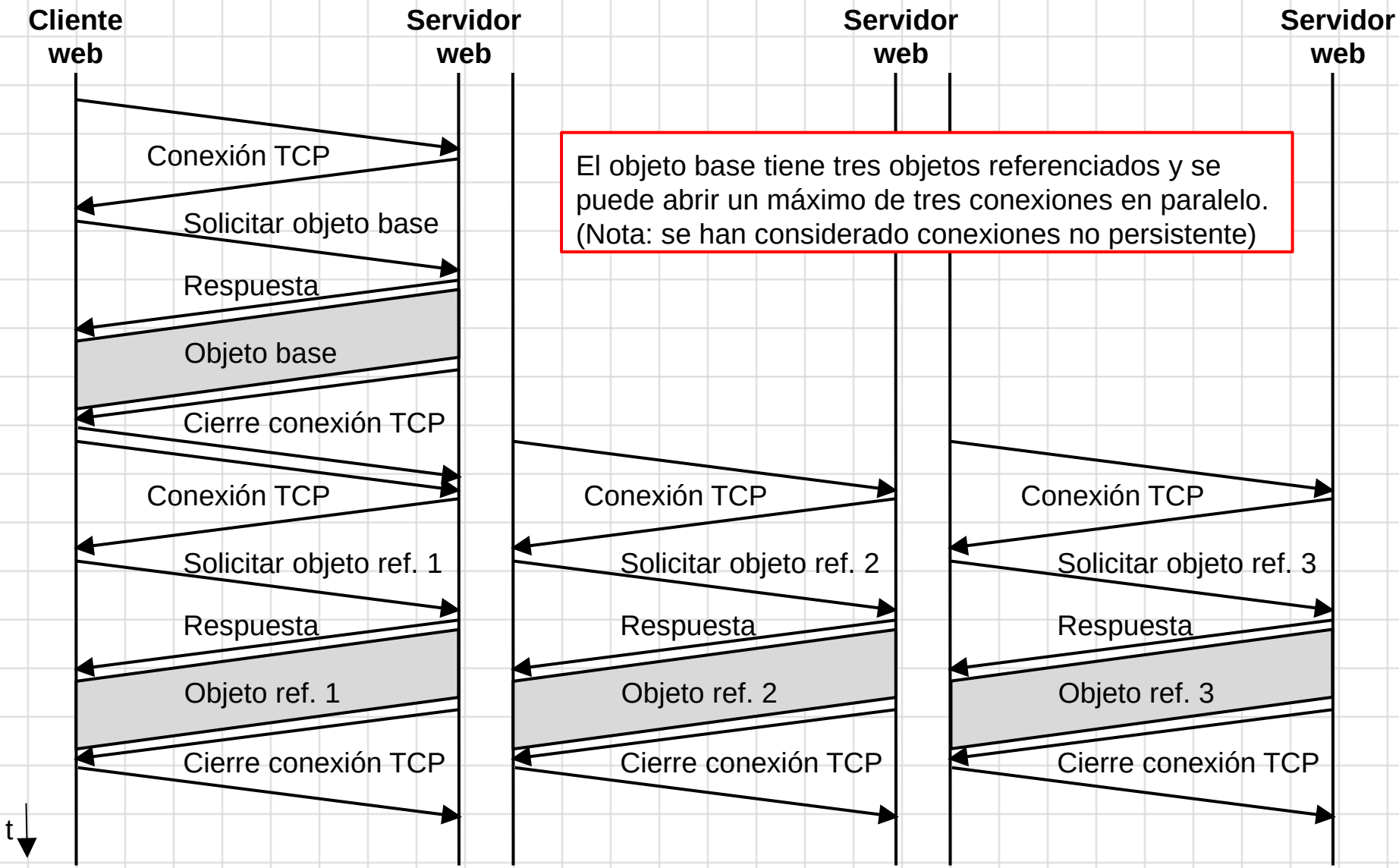


Tiempo de respuesta- Transparencia 37





Conexiones en paralelo - Transparencia 38



TR = 2RTT + TTO + 2RTT + TTO

¿Cuál es el tiempo de respuesta sin las conexiones en paralelo?

Boletín Tema 2 - P3

a) De las dos primeras líneas del mensaje:

GET /cs453/index.html HTTP/1.1←↓
Host: gaia.cs.umass.edu←↓

puedo extraer la HTTP_UD que corresponde con la URL del documento solicitado:

gaia.cs.umass.edu/cs453/index.html

b) De la primera línea del mensaje (línea de petición):

GET /cs453/index.html HTTP/1.1←↓

puedo extraer la versión de HTTP ejecutada en el navegador: 1.1

c) Para comprobar si se solicita una conexión persistente hay que fijarse en la línea de cabecera “Connection”:

Connection: keep-alive←↓

Se solicita una conexión persistente ya que está establecida la opción “keep-alive”

d) La dirección IP del host que ejecuta el navegador web no se incluye en la PCI de la HTTP_PDU de petición. Esta información va en la PCI de la IP_PDU.

Boletín Tema 2 - P3

e) Para conocer el tipo de navegador que envía la petición hay que fijarse en la línea de cabecera “User-Agent”:

User-Agent: Mozilla/5.0 (Windows;U; Windows NT 5.1; en-US; rv:1.7.2) … ←↓

La petición se está haciendo desde un navegador Mozilla. Es necesario indicar el tipo de navegador ya que el servidor puede disponer de varias versiones del mismo objeto, cada una adaptada a un tipo de navegador.

f) El tamaño de la HTTP_PDU se calcula sumando todos los caracteres enviados: 411 bytes.

g) El tamaño de la HTTP_UD corresponde al tamaño de la URL: 34 bytes.

Boletín Tema 2 - P4

a) De la primera línea del mensaje:

HTTP/1.1 200 OK←↓

puedo extraer el código y el mensaje de estado. En este caso indica que lo ha encontrado. Para saber cuando lo ha suministrado nos fijamos en la línea de cabecera “Date”:

Date: Tue, 07 Mar 2008 12:39:45 GMT←↓

b) La línea de cabecera “Last-modified” nos proporciona la fecha de última modificación:

Last-modified: Sat, 10 Dec 2005 18:27:46 GMT←↓

c) El documento devuelto se corresponde con la página web (HTTP_UD). La línea de cabecera “Content-Length” proporciona esta información:

Content-Length: 3874←↓

d) El final de la HTTP_PCI viene delimitado por la aparición de los caracteres \r\n (representados como ←↓ en la cadena) al principio de una línea:

Content-Type: text/html; charset=ISO-8859-1←↓
←↓
<!doctype html

Los cinco primeros bytes del documento serían: <!doc

Boletín Tema 2 - P4

e) Para comprobar si se ha acordado emplear una conexión persistente nos fijamos en la línea de cabecera “Connection”:

Connection: keep-alive←↓

Se acuerda una conexión persistente ya que está establecida la opción “keep-alive”. Para conocer el tiempo máximo de inactividad nos fijamos en la línea de cabecera “Keep-Alive”:

Keep-Alive: timeout=15, max=100←↓

El servidor cerrará la conexión persistente tras 15 segundos de inactividad o tras solicitarse 100 objetos por dicha conexión.

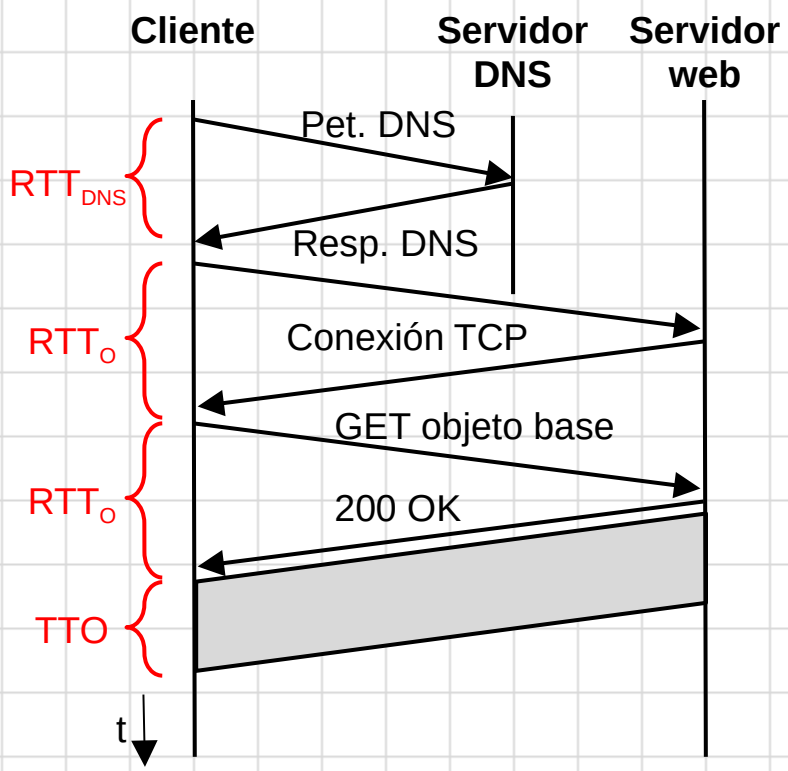
f) La HTTP_UD que envía el servidor se corresponde con la página web. Por tanto, es similar al apartado c.

g) El tamaño de la HTTP_PDU sería el resultado de sumar el tamaño de la HTTP_PCI y el tamaño de la HTTP_UD (ya conocido). Para calcular el tamaño de la HTTP_PCI hay que contar todos los caracteres de la cadena hasta que comienza el documento (revisar apartado d).

Tamaño HTTP_PCI = 308 bytes.
Tamaño HTTP_UD = 3874 bytes.

Tamaño HTTP_PDU = 308 + 3874 = 4182 bytes.

Boletín Tema 2 - P5

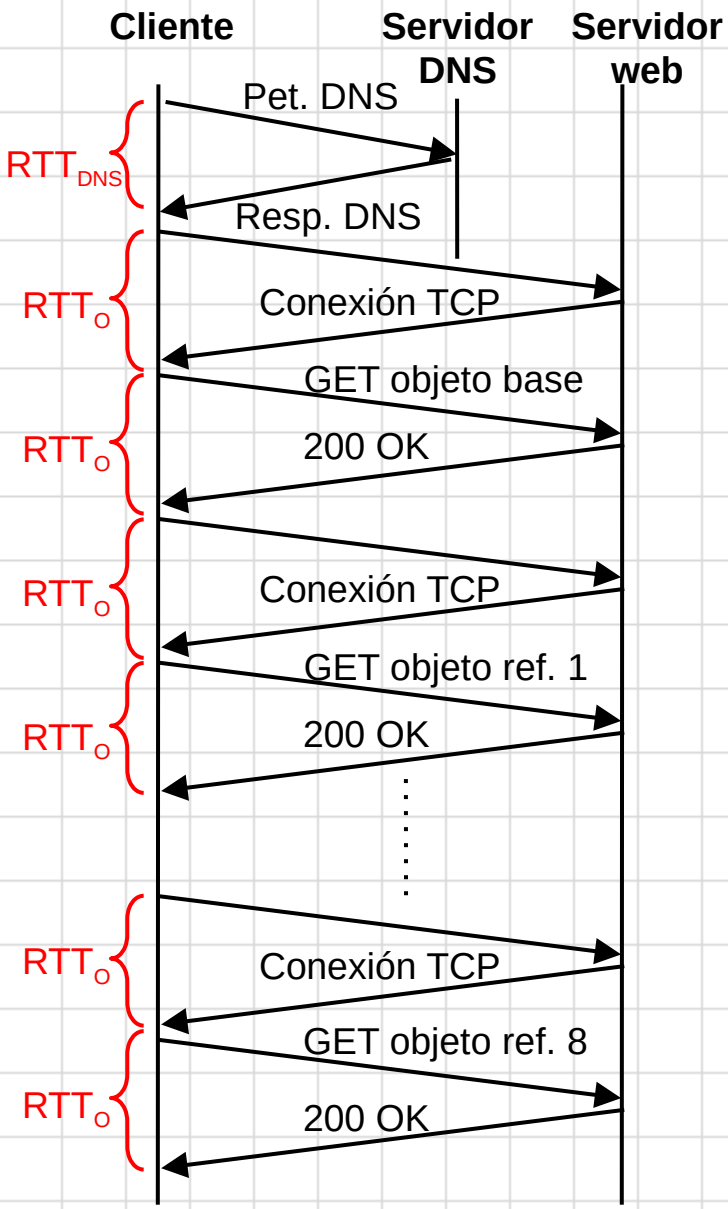


$TR = RTT_{DNS} + 2RTT_O + TTO$

De acuerdo al enunciado la página web asociada con el vínculo es un pequeño fichero HTML lo que supone un tiempo de transmisión del objeto despreciable (TTO = 0 s) . Por tanto, el tiempo de respuesta final es:

$TR = RTT_{DNS} + 2RTT_O$

Boletín Tema 2 - P6, ap. a)



Tiempo de respuesta usado en resolver el nombre y solicitar el objeto base (problema 5):

$$TR_{DNS+O.BASE} = RTT_{DNS} + 2RTT_O$$

Tiempo de respuesta empleado en la descarga de un objeto usando conexiones no persistentes:

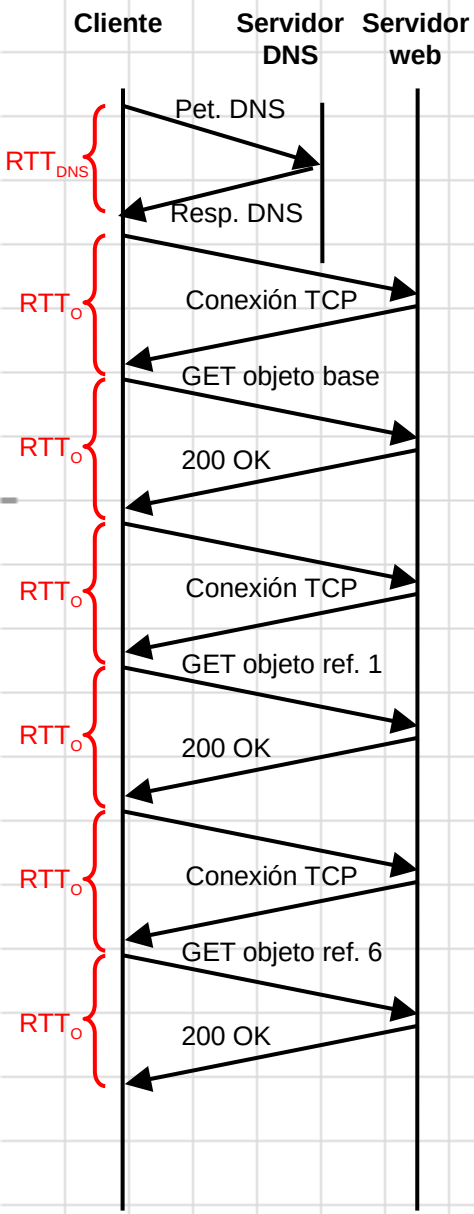
$$TR_{O.REF} = 2RTT_O$$

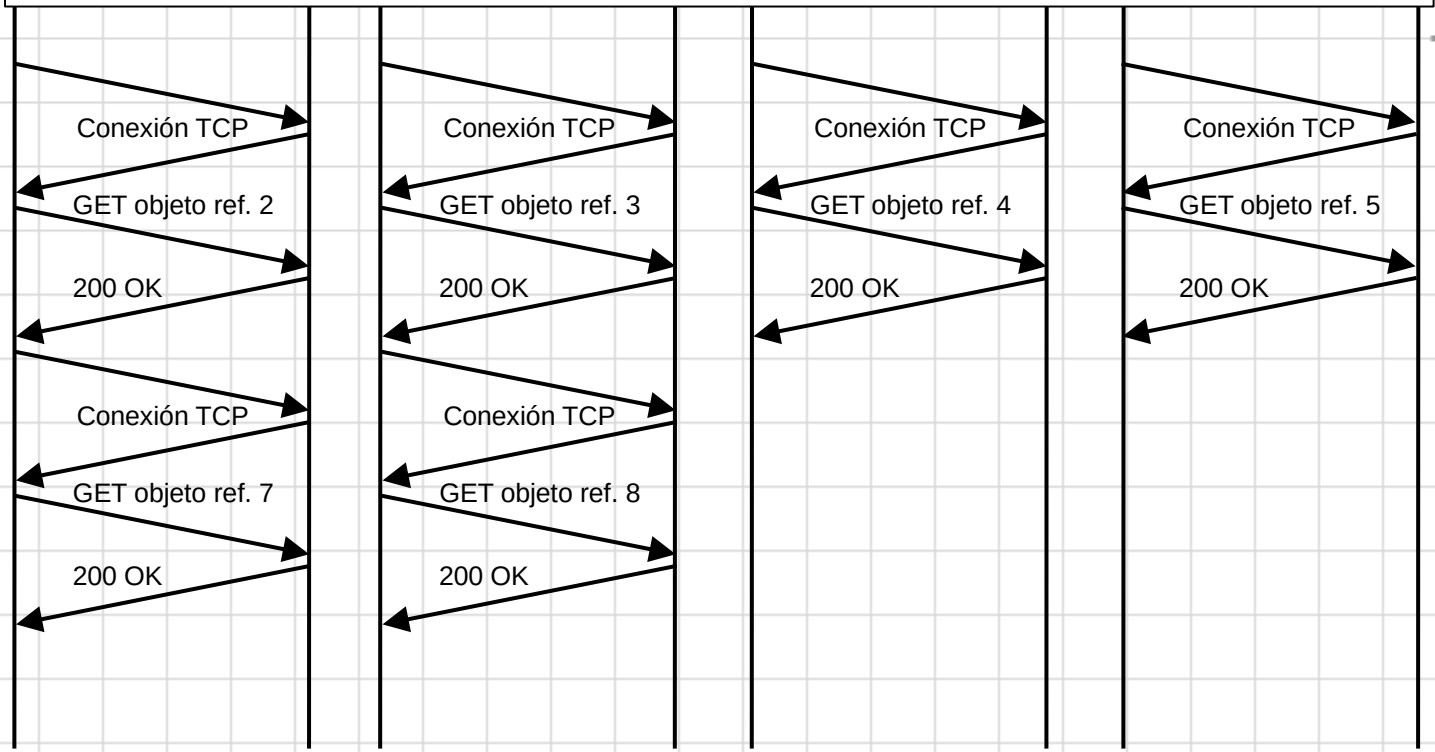
Como hay 8 objetos referenciados el tiempo de respuesta final es:

$$TR = RTT_{DNS} + 2RTT_O + 8(2RTT_O) = RTT_{DNS} + 18RTT_O$$

De acuerdo al enunciado el tamaño de los objetos es muy pequeño por lo que consideramos que el TTO es despreciable.

Boletín Tema 2 - P6, ap. b)





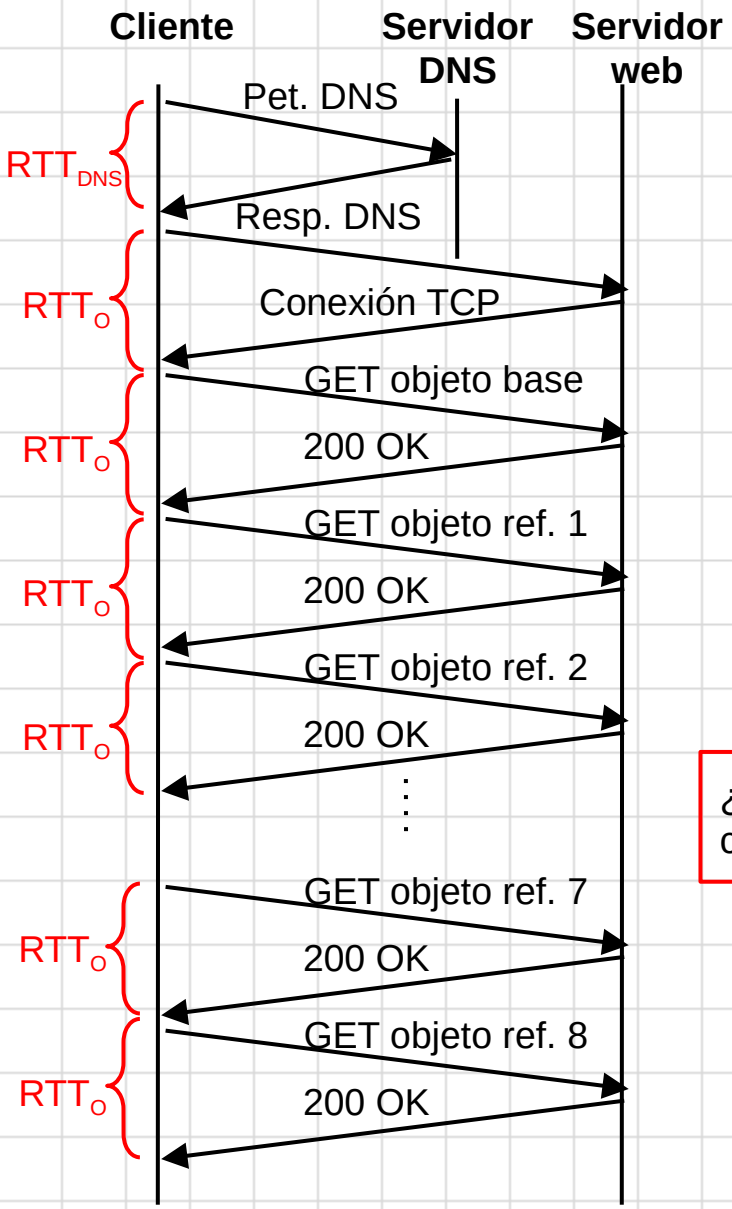
Tiempo de respuesta usado en resolver el nombre y solicitar el objeto base (problema 5):

$$TR_{DNS+O.BASE} = RTT_{DNS} + 2RTT_O$$

Tiempo de respuesta empleado en la descarga de los 8 objetos referenciados usando conexiones no persistentes y 5 conexiones en paralelo:

$$TR = RTT_{DNS} + 2RTT_O + 4RTT_O = RTT_{DNS} + 6RTT_O$$

Boletín Tema 2 - P6, ap. c)



Tiempo de respuesta usado en resolver el nombre y solicitar el objeto base (problema 5):

$$TR_{DNS+O.BASE} = RTT_{DNS} + 2RTT_O$$

Tiempo de respuesta empleado en la descarga de un objeto usando conexiones no persistentes:

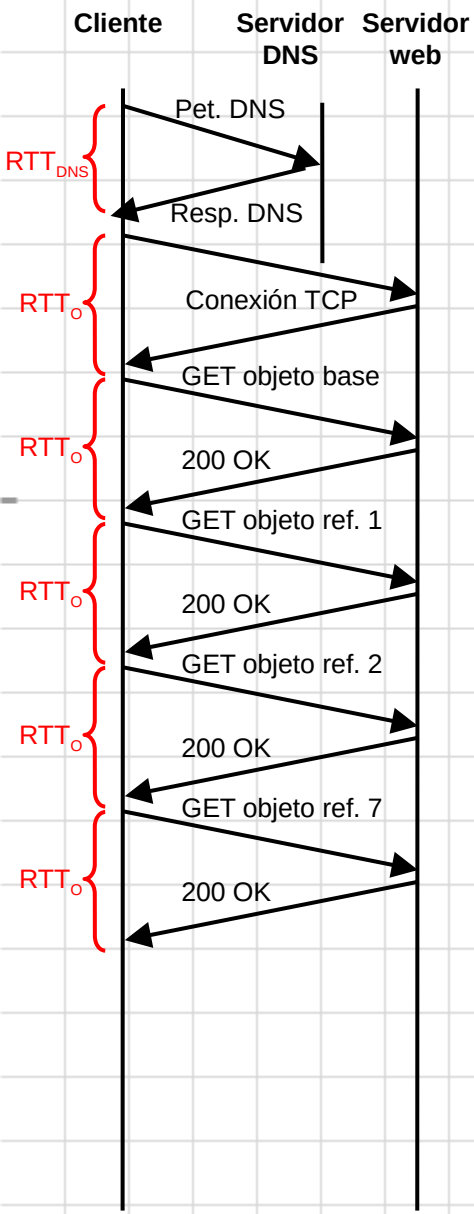
$$TR_{O.REF} = RTT_O$$

Como hay 8 objetos referenciados el tiempo de respuesta final es:

$$TR = RTT_{DNS} + 2RTT_O + 8RTT_O = RTT_{DNS} + 10RTT_O$$

¿Cuánto tiempo transcurre si se utiliza HTTP persistente con 5 conexiones en paralelo?

Boletín Tema 2 - P6, ap. d)

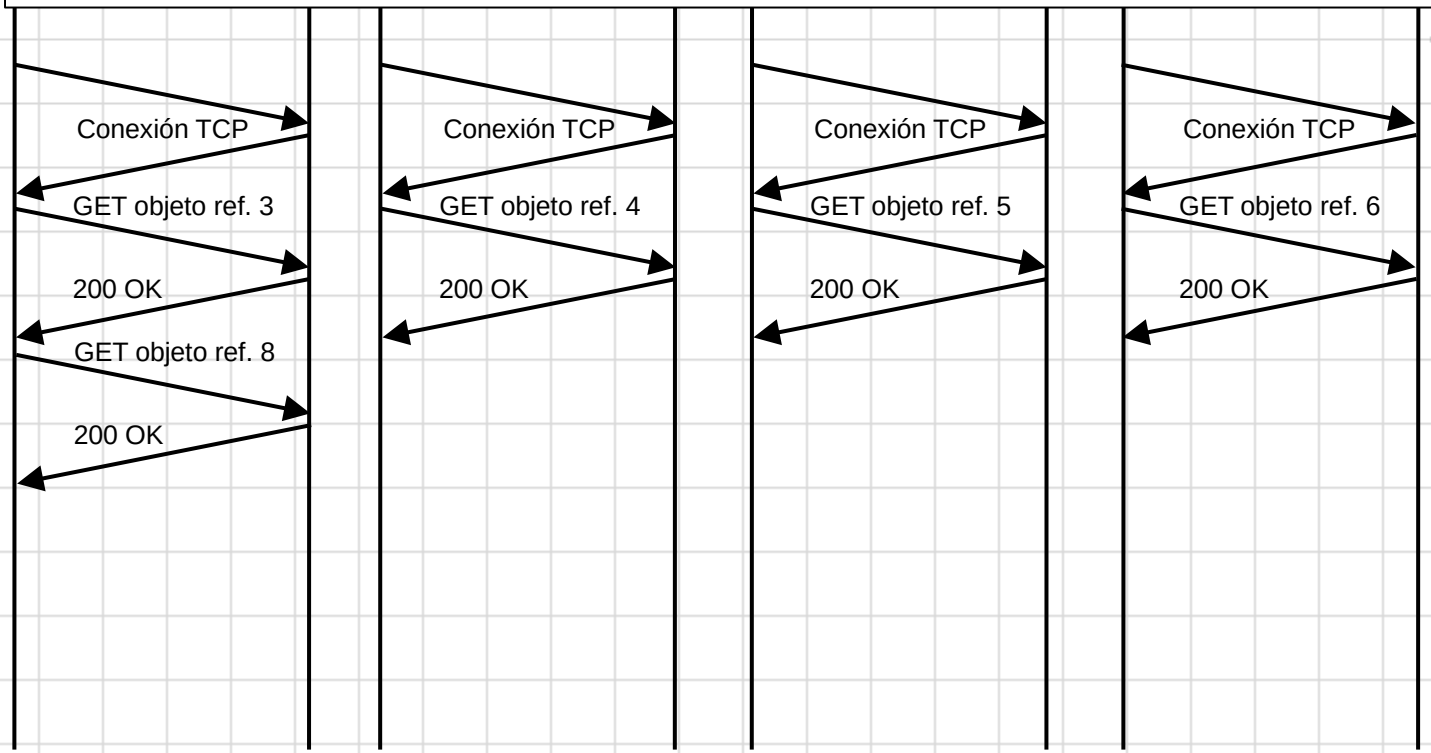


Tiempo de respuesta usado en resolver el nombre y solicitar el objeto base (problema 5):

$$TR_{DNS+O.BASE} = RTT_{DNS} + 2RTT_O$$

Tiempo de respuesta empleado en la descarga de los 8 objetos referenciados usando conexiones persistentes y 5 conexiones en paralelo:

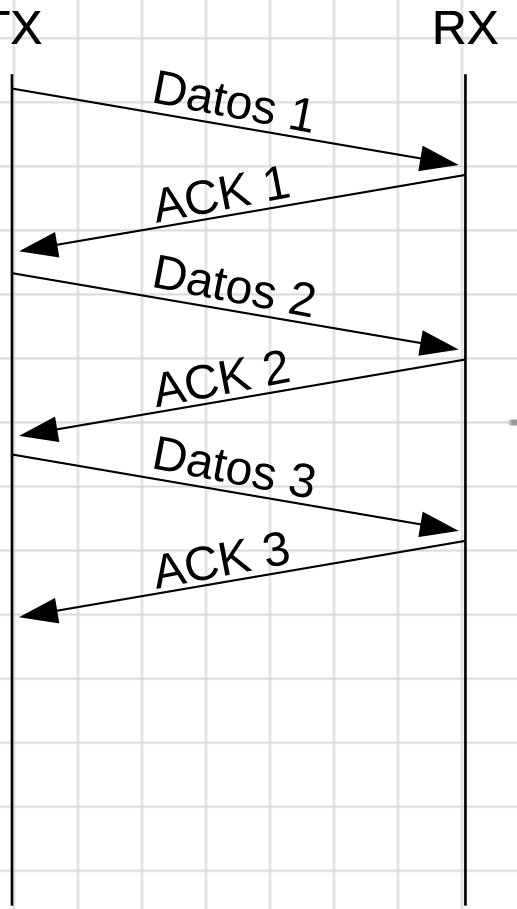
$$TR = RTT_{DNS} + 2RTT_O + 3RTT_O = RTT_{DNS} + 5RTT_O$$



Problemas
Tema 3

Problema transferencia fiable (I) (no está en el boletín)

Considere que se ha empleado un protocolo de transferencia fiable (PTF) para enviar las tres PDUs de datos mostradas en la Figura. Como se observa en esta figura no se han producido errores de bit ni pérdidas de PDU durante la comunicación de los datos. En relación con las características del protocolo utilizado, el PTF implementa un mecanismo de parada y espera y numera las PDU de datos y de control empezando por el número de secuencia 1. Además, este protocolo no implementa el acuse de recibo negativo (NACK) pero sí la comprobación de errores de bit (*checksum*) y un temporizador que permite detectar si ha expirado el *timeout*. Este temporizador comienza su operación tras enviar una PDU de datos.

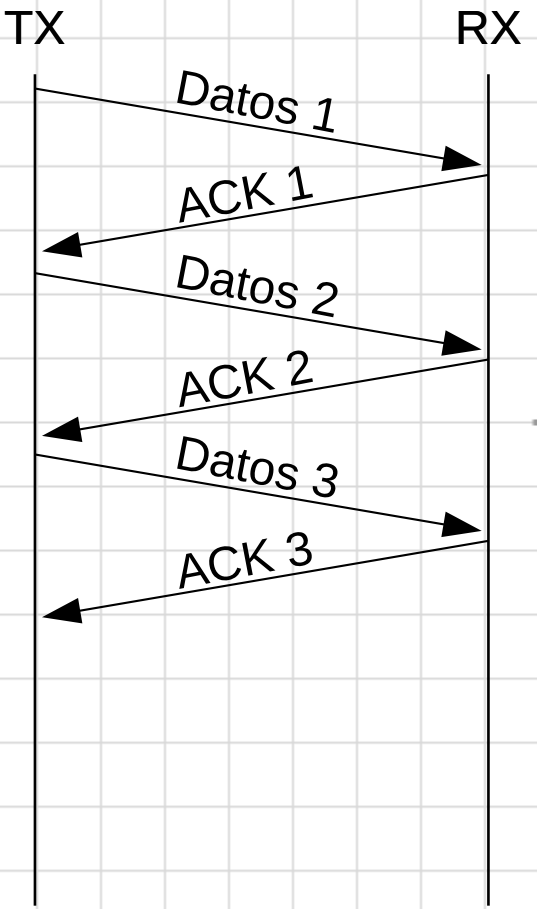


Responda de manera razonada las siguientes preguntas:

1) Suponiendo que el receptor ha detectado errores de bit al recibir la segunda PDU de datos, represente la comunicación que se produce entre el transmisor y el receptor para enviar todas las PDU de manera fiable. Si comparamos la representación realizada con la transferencia sin errores, ¿qué diferencias existen? ¿Qué nombre recibe la PDU de control enviada por el receptor para informar que se han producido errores de bit?

Problema transferencia fiable (II) (no está en el boletín)

Considere que se ha empleado un protocolo de transferencia fiable (PTF) para enviar las tres PDUs de datos mostradas en la Figura. Como se observa en esta figura no se han producido errores de bit ni pérdidas de PDU durante la comunicación de los datos. En relación con las características del protocolo utilizado, el PTF implementa un mecanismo de parada y espera y numera las PDU de datos y de control empezando por el número de secuencia 1. Además, este protocolo no implementa el acuse de recibo negativo (NACK) pero sí la comprobación de errores de bit (*checksum*) y un temporizador que permite detectar si ha expirado el *timeout*. Este temporizador comienza su operación tras enviar una PDU de datos.



Responda de manera razonada las siguientes preguntas:

1) Represente la comunicación de datos fiable entre el transmisor y el receptor suponiendo que:
   1) Se ha perdido el primer ACK representado en la Figura.
   2) Se ha perdido la segunda PDU de datos representada en la Figura.
   3) El último ACK representado en la Figura ha llegado con errores al transmisor.

(Nota: debe indicar en el diagrama qué PDU llegan duplicadas al receptor)

Problemas
Tema 4

Dir. sin clase - Transparencia 44

El sistema final con dirección IP 223.1.12.2 quiere enviar una IP_PDU al sistema final con dirección IP 223.1.8.2. Para ello, consulta su tabla de enrutamiento empezando por las entradas con mayor prefijo (/22 en este ejemplo) y terminando, si es el caso, por las de menor (/0).
Una vez aplicada la máscara asociada al prefijo de red de una entrada debe compararse la dirección de red resultante con la dirección de red de esa entrada. Si coinciden debe aplicarse esa entrada. En otro caso hay que seguir operando. Si no se puede aplicar ninguna entrada la IP_PDU se descarta.

Dirección IP destino	223.1.8.2
Máscara de subred a aplicar	255.255.252.0
Dirección de red (tras aplicar la máscara)	223.1.8.0

He aplicado la máscara de la primera entrada que corresponde con la red lógica 223.1.12.0 (a la que pertenece el sistema final origen). Las direcciones de red no coinciden (223.1.12.0 ≠ 223.1.8.0) por lo que no puedo aplicar esta entrada. Hay que seguir operando.

1º Byte	2º Byte	3º Byte	4º Byte
(223) 11011111	(1) 00000001	(8) 00001000	(2) 00000010
(255) 11111111	(255) 11111111	(252) 11111100	(0) 00000000
(223) 11011111	(1) 00000001	(8) 00001000	(0) 00000000

Dir. sin clase - Transparencia 44

Como es la tabla de enrutamiento de un sistema final sólo queda por consultar la ruta por defecto. Al aplicar la máscara de la ruta por defecto obtenemos la dirección de red 0.0.0.0 que coincide con la red contenida en esa entrada. Consultando la columna próximo salto se determina el router al que hay que enviar la IP_PDU.

Dirección IP destino	223.1.8.2
Máscara de subred a aplicar	0.0.0.0
Dirección de red (tras aplicar la máscara)	0.0.0.0

1º Byte	2º Byte	3º Byte	4º Byte
(223) 11011111	(1) 00000001	(8) 00001000	(2) 00000010
(0) 00000000	(0) 00000000	(0) 00000000	(0) 00000000
(0) 00000000	(0) 00000000	(0) 00000000	(0) 00000000

Dir. sin clase - Transparencia 44

Una vez que la IP_PDU llega al router debemos consultar su tabla de enrutamiento para determinar la interfaz por la que debe reenviarse. Como todas tienen el mismo prefijo (/22) decido consultarlas empezando por la primera entrada con dirección de red 223.1.4.0. Al aplicar la máscara de la primera entrada se observa que las direcciones de red no coinciden (223.1.4.0 ≠ 223.1.8.0) por lo que no puedo aplicar esa entrada. Hay que seguir operando.

Dirección IP destino	223.1.8.2
Máscara de subred a aplicar	255.255.252.0
Dirección de red (tras aplicar la máscara)	223.1.8.0

1º Byte	2º Byte	3º Byte	4º Byte
(223) 11011111	(1) 00000001	(8) 00001000	(2) 00000010
(255) 11111111	(255) 11111111	(252) 11111100	(0) 00000000
(223) 11011111	(1) 00000001	(8) 00001000	(0) 00000000

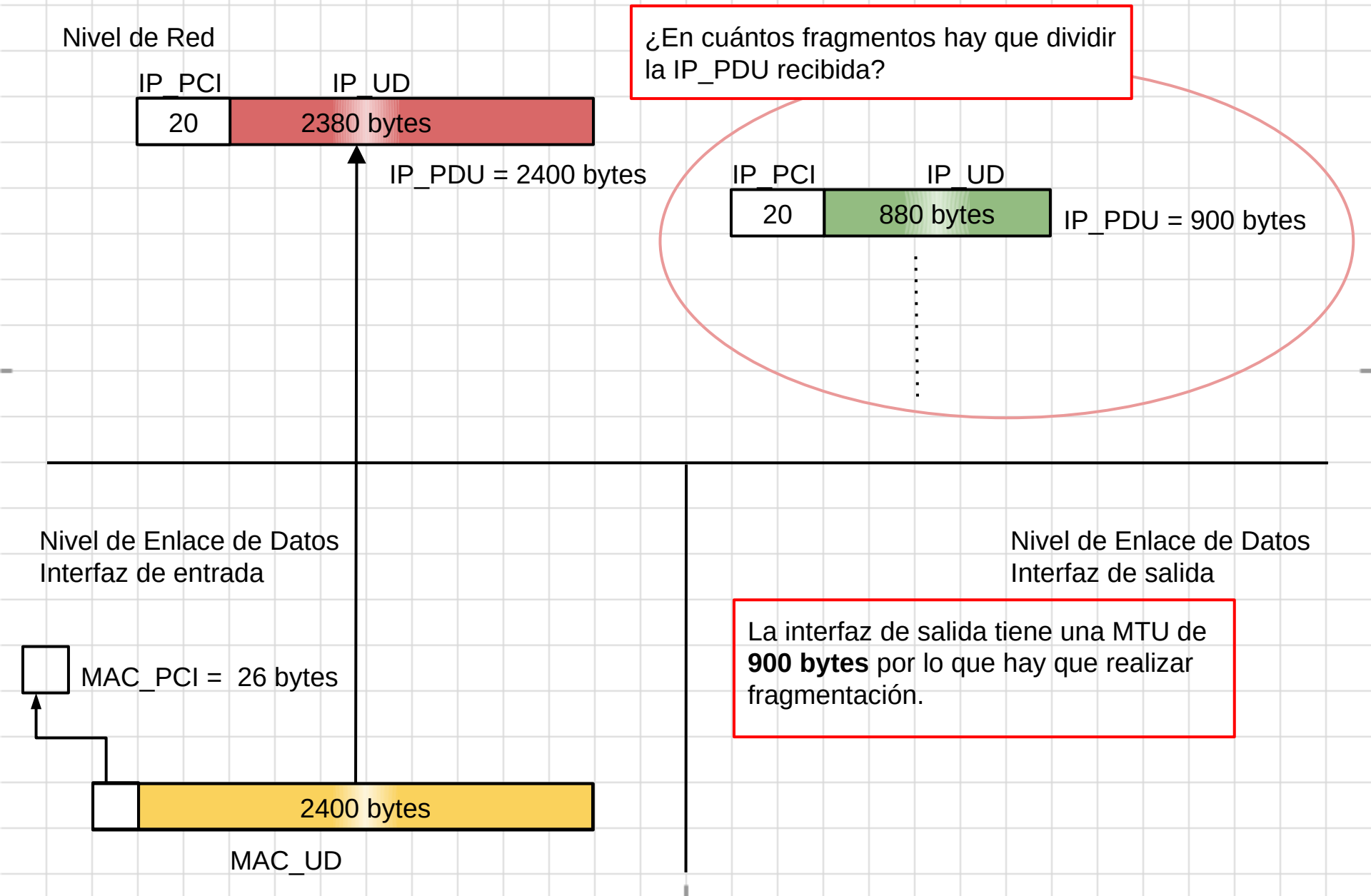
Dir. sin clase - Transparencia 44

Consultamos la segunda entrada con dirección de red 223.1.8.0. Al aplicar la máscara de la segunda entrada se comprueba que las direcciones de red coinciden (223.1.8.0 = 223.1.8.0); esto significa que hay que actuar de acuerdo al próximo salto contenido en esta entrada. En esta ocasión el próximo salto nos informa que el dispositivo de destino está directamente conectado por la interfaz E1. Por tanto, el router puede comunicarse con el destino (sistema final con dirección IP 223.1.8.2) directamente a través de esta interfaz y reenviarle la IP_PDU.

Dirección IP destino	223.1.8.2
Máscara de subred a aplicar	255.255.252.0
Dirección de red (tras aplicar la máscara)	223.1.8.0

1º Byte	2º Byte	3º Byte	4º Byte
(223) 11011111	(1) 00000001	(8) 00001000	(2) 00000010
(255) 11111111	(255) 11111111	(252) 11111100	(0) 00000000
(223) 11011111	(1) 00000001	(8) 00001000	(0) 00000000

Boletín Tema 4 – P1 (Modificado)



Problemas Examen

Curso 2011/12

Ev. Alternativa - P1 ap. a) y b)

Apartado a) Considerando que el Rext no pertenece a la empresa y por tanto el dominio de broadcast entre Rext y R1 tampoco, en la red de la empresa hay dos dominios de broadcast. La interfaz E0 de R1 pertenece al primero y la interfaz E1 al segundo.

Apartado b) Considerando que el bloque CIDR que tiene asignado la red de la empresa es el 200.1.1.0/24 y que dicho bloque no está dentro del rango reservado para direccionamiento privado, la entrada de la tabla de enrutamiento de Rext que sirve para reenviar el tráfico hacia la red de la empresa es la siguiente:

T.E. Rext

Red	Próximo Salto	Interfaz
200.1.1.0/24	150.214.141.24	E7

Apartado b) Es muy importante la consideración de que no sea un bloque privado. Si no es privado el Rext obtendrá información de enrutamiento de la red de la empresa, añadiendo una entrada en su TE con la información que permite alcanzar esa red.

Ev. Alternativa - P1 ap. c)

Apartado c) En primer lugar vamos a determinar cuántas direcciones IP necesitamos en cada subred:

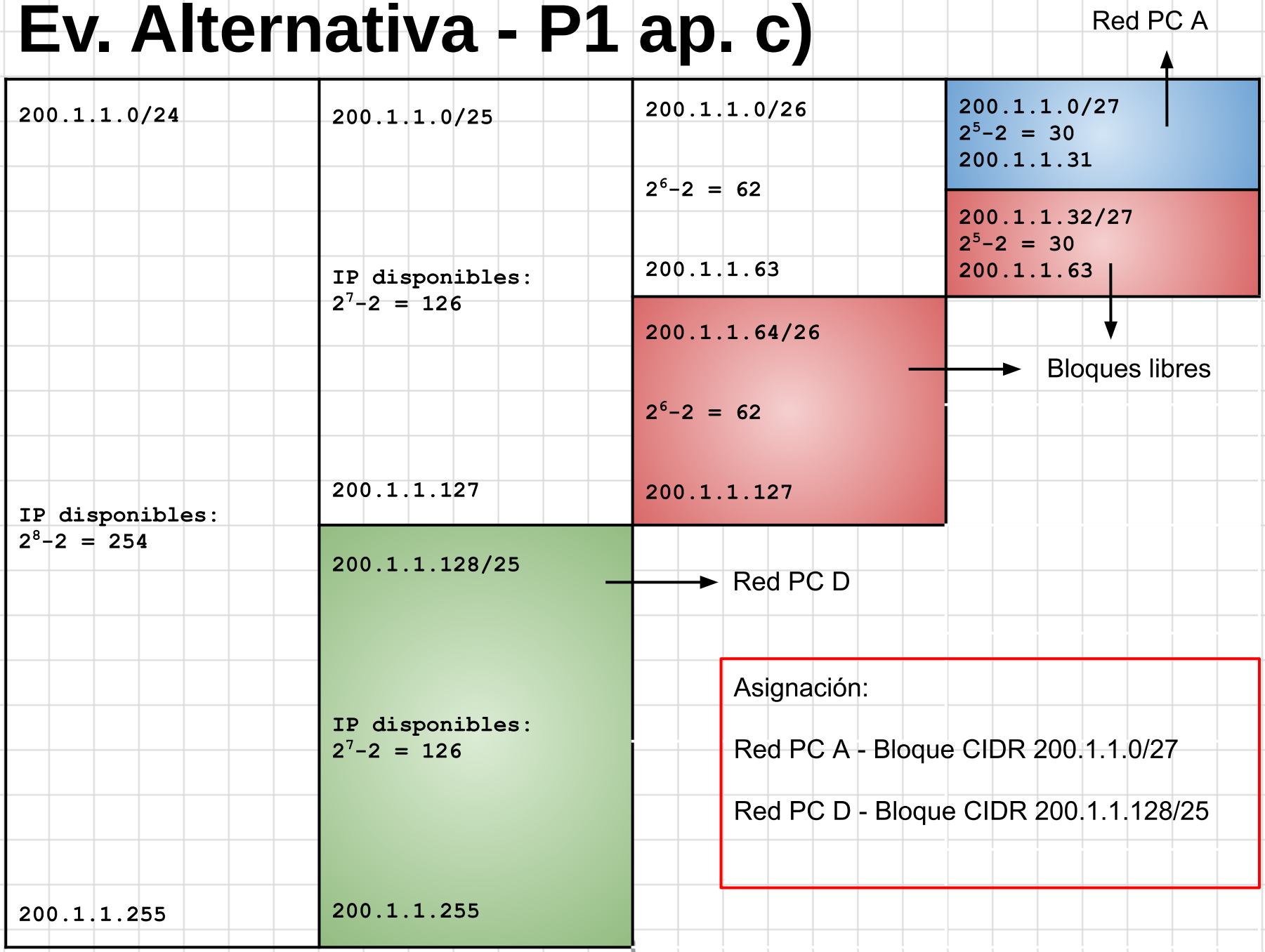
- **Subred PC A**: hay 14 sistemas finales. Considerando que a la interfaz E0 del router R1 también hay que asignarle una dirección IP necesitamos **15 direcciones IP**.
- **Subred PC D**: hay 90 sistemas finales y la interfaz E1 de R1. Por tanto, necesitamos **91 direcciones IP**.

A continuación, vamos a calcular los prefijos de red considerando que hay que dejar sin asignar el mayor número de IP para futuras ampliaciones de la red:

- **Subred PC A**: con un prefijo /28 dedicamos 28 bits para la red y 4 bits para la parte de host. Con 4 bits se pueden direccionar 2^4 - 2 = 14 dispositivos con nivel de red (sistemas finales y routers) por lo que no tenemos suficientes direcciones. Necesitamos asignar por tanto un prefijo /27.
- **Subred PC D**: con un prefijo /25 dedicamos 25 bits para la red y 7 bits para la parte de host. Con 7 bits se pueden direccionar 2^7 - 2 = 126 dispositivos con nivel de red (sistemas finales y routers) por lo que incluso se desaprovecharán algunas direcciones IP. Asignamos por tanto un prefijo /25.

La asignación final de prefijos se muestra en la tabla de la siguiente transparencia.

Ev. Alternativa - P1 ap. c)



Ev. Alternativa - P1 ap. d)

Configuración IPv4 de:	Dirección IP asignada	Máscara de subred	Dirección IP Router Frontera
PC A	200.1.1.2	255.255.255.224	200.1.1.1
PC D	200.1.1.130	255.255.255.128	200.1.1.129

T.E. R1

Red	Próximo Salto	Interfaz
200.1.1.0/27	-	E1
200.1.1.128/25	-	E0
Dirección red/prefijo	-	E2
0.0.0.0/0	IP Rext - E7	E2

Queda por determinar: 1) la dirección de red y el prefijo de la red directamente conectada por la interfaz E2, y 2) la dirección del próximo salto de la ruta por defecto. Para ello, debemos partir de la configuración IPv4 asignada a la interfaz E2 de R1: 150.214.141.24/27. En notación máscara:

- IP: 150.214.141.24
- Máscara de subred: 255.255.255.224

Ev. Alternativa - P1 ap. d)

T.E. R1

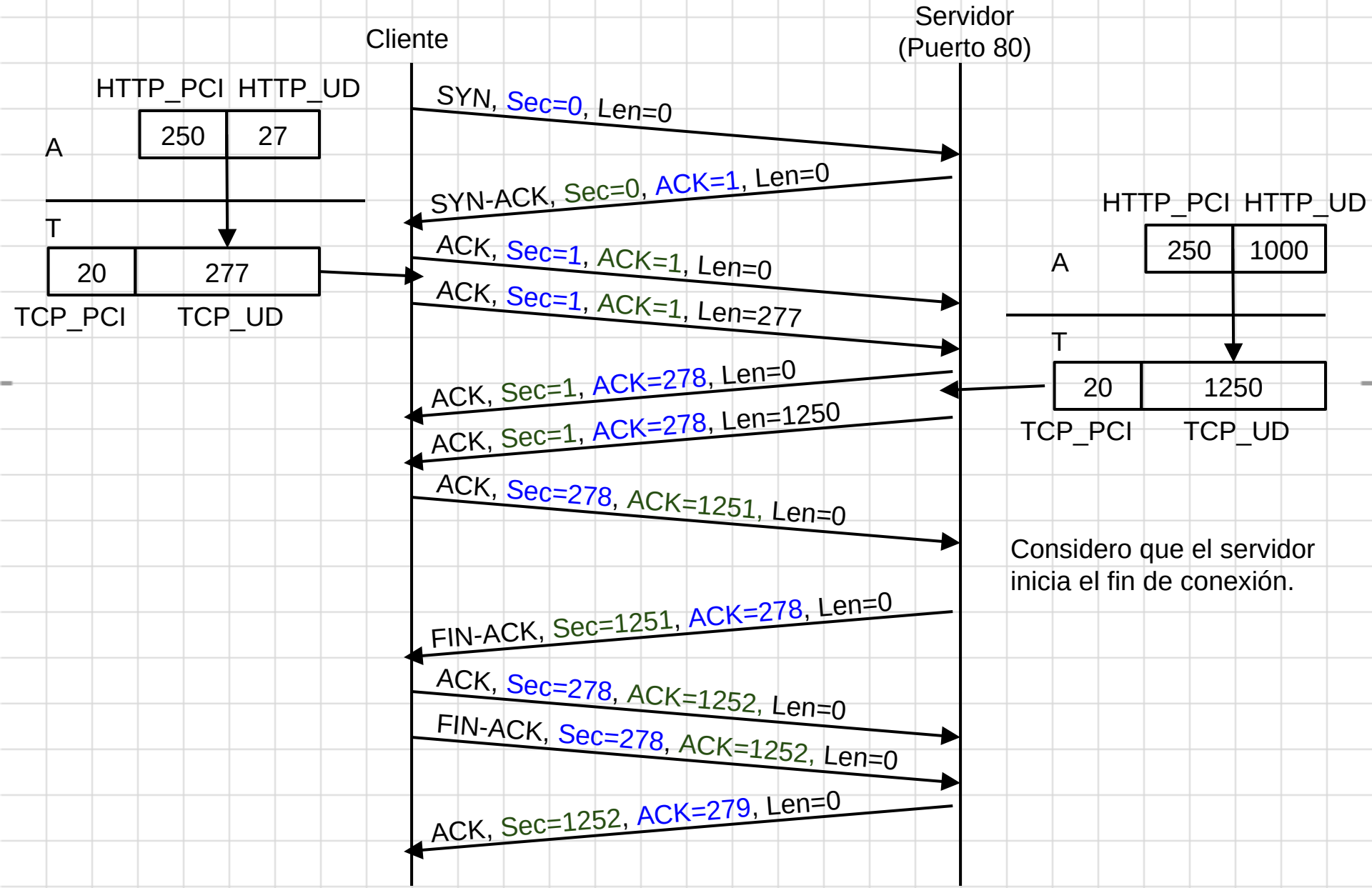
Red	Próximo Salto	Interfaz
200.1.1.0/27	-	E1
200.1.1.128/25	-	E0
150.214.141.0/27	-	E2
0.0.0.0/0	150.214.141.25	E2

Para determinar la dirección de red aplicamos la máscara a la dirección IP asignada a la interfaz:

1º Byte	2º Byte	3º Byte	4º Byte
(150) 10010110	(214) 11010110	(141) 10001101	(24) 00011000
(255) 11111111	(255) 11111111	(255) 11111111	(224) 11100000
(150) 10010110	(214) 11010110	(141) 10001101	(0) 00000000

Para determinar la dirección IP del próximo salto de la ruta por defecto hay que analizar las direcciones asignables dentro de ese bloque. En este caso, como no se proporciona información de ningún otro dispositivo conectado en esa red podemos considerar que las direcciones IP libres están en los rangos: 150.214.141.1 a 150.214.141.23 y 150.214.141.25 a 150.214.141.30. Puede elegir cualquier IP en esos rangos.

Ev. Alternativa - P3



Febrero 2012 - P1 ap. a) y b)

Apartado 1) Considerando que el Rext no pertenece a la empresa y por tanto el dominio de broadcast entre Rext y R1 tampoco, en la red de la empresa hay tres dominios de broadcast. Las interfaces Fa0 de R1 y R2 pertenecen al primero, la interfaz Fa1 de R2 al segundo y la interfaz Fa2 de R2 al tercero. Faltaría indicar los dominios de colisión (cuando se explique el tema 5).

Apartado b) Considerando que el router R1 implementa NAT debemos considerar nuestra red como una red privada. Esto significa que el Rext no obtendrá información de enrutamiento de la red la empresa y que todos los datagramas que reenvíe R1 hacia Internet saldrán con dirección IP origen 150.214.141.25. Por tanto, cuando Rext reciba un datagrama procedente de Internet destinado a la red de la empresa la dirección IP destino será 150.214.141.25. Esto implica que la información de enrutamiento que debe disponer Rext para reenviar el datagrama hacia la red de la empresa es aquella que permita la comunicación entre los dispositivos de la red que conecta R1 con Rext.

T.E. Rext

Red	Próximo Salto	Interfaz
150.214.141.24/30	-	I9

Febrero 2012 - P1 ap. c)

Apartado c) En primer lugar vamos a determinar cuántas direcciones IP necesitamos en cada subred:

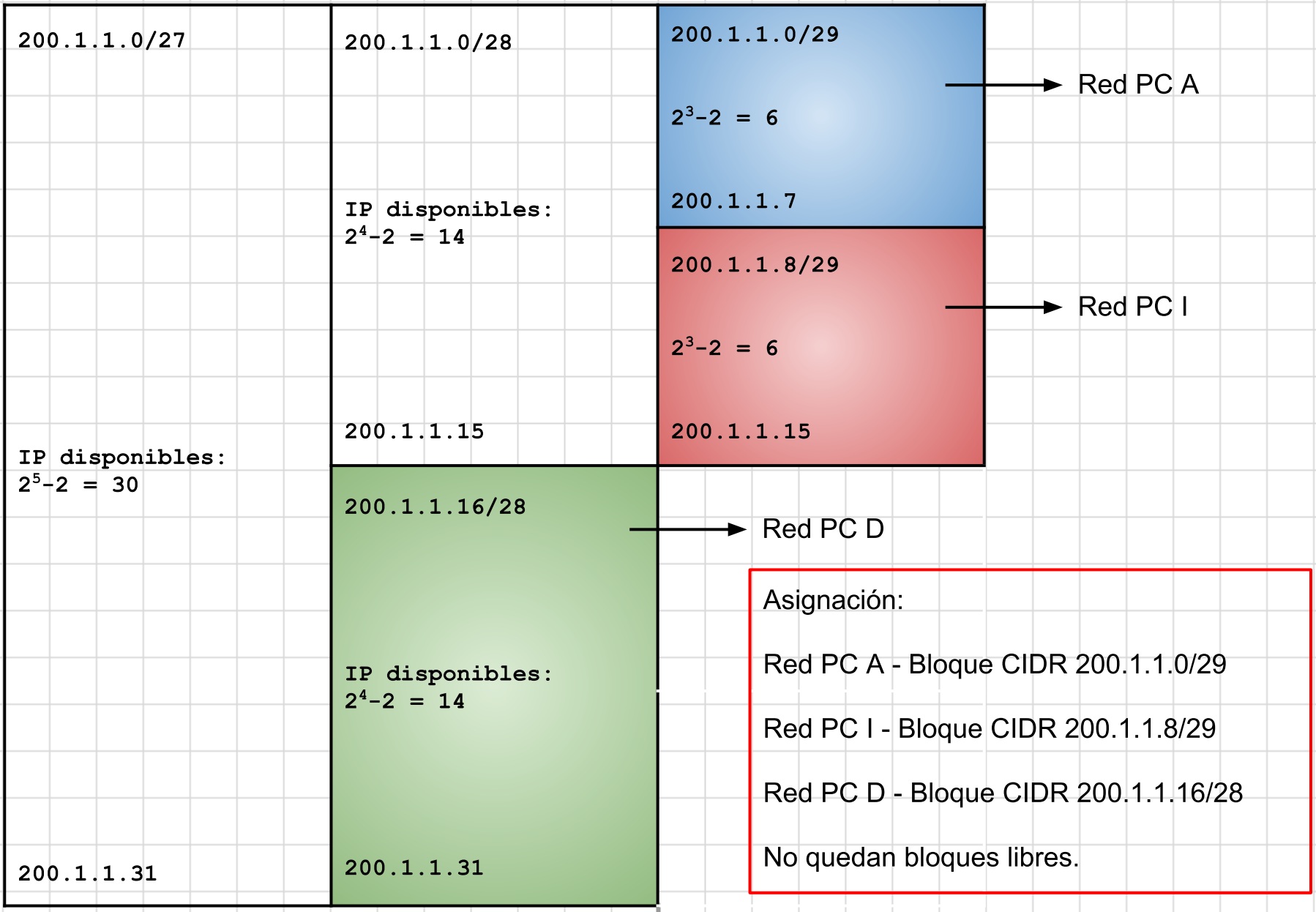
- **Subred PC A**: hay 4 sistemas finales y la interfaz Fa1 de R2. Necesitamos **5 direcciones IP**.
- **Subred PC D**: hay 5 sistemas finales y las interfaces Fa0 de R1 y R2. Necesitamos **7 direcciones IP**.
- **Subred PC I:** hay 4 sistemas finales y la interfaz Fa2 de R2. Necesitamos **5 direcciones IP**.

A continuación, vamos a calcular los prefijos de red:

- **Subred PC A**: con un prefijo /29 dedicamos 29 bits para la red y 3 bits para la parte de host. Con 3 bits se pueden direccionar 2^3 - 2 = 6 dispositivos con nivel de red (sistemas finales y routers) por lo que tenemos suficientes direcciones. Necesitamos asignar por tanto un prefijo /29.
- **Subred PC D**: con un prefijo /29 no tendríamos suficientes (ver asignación de la subred PC A). Necesitamos un /28. Con un prefijo /28 dedicamos 28 bits para la red y 4 bits para la parte de host. Con 4 bits se pueden direccionar 2^4 - 2 = 14 dispositivos con nivel de red (sistemas finales y routers).
- **Subred PC I**: es similar a la subred PC A. Necesitamos un prefijo /29.

La asignación final de prefijos se muestra en la tabla de la siguiente transparencia.

Febrero 2012 - P1 ap. c)



Asignación:

Red PC A - Bloque CIDR 200.1.1.0/29

Red PC I - Bloque CIDR 200.1.1.8/29

Red PC D - Bloque CIDR 200.1.1.16/28

No quedan bloques libres.

Febrero 2012 - P1 ap. d)

Configuración IPv4 de:	Dirección IP asignada	Máscara de subred	Dirección IP Router Frontera
PC A	200.1.1.2	255.255.255.248	200.1.1.1
PC D	200.1.1.19	255.255.255.240	200.1.1.17
PC I	200.1.1.10	255.255.255.248	200.1.1.9

Nota: he asignado la dirección IP 200.1.1.17 a la interfaz Fa0 de R1 y la dirección IP 200.1.1.18 a la interfaz Fa0 de R2. Considero que el router frontera de los sistemas finales de la subred del PC D es R1.

T.E. R1

Red	Próximo Salto	Interfaz
200.1.1.16/28	-	Fa0
150.214.141.24/30	-	I0
~~200.1.1.0/29~~	~~200.1.1.18~~	~~Fa0~~
~~200.1.1.8/29~~	~~200.1.1.18~~	~~Fa0~~
0.0.0.0/0	150.214.141.26	I0

Se pide una tabla de enrutamiento de tamaño mínimo. Como esas dos subredes se han creado a partir del bloque CIDR 200.1.1.0/28 y el próximo salto es el mismo es posible proporcionar la misma información con una sola entrada:

200.1.1.0/28	200.1.1.18	Fa0

Febrero 2012 - P1 ap. d)

T.E. R1

Red	Próximo Salto	Interfaz
200.1.1.16/28	-	Fa0
150.214.141.24/30	-	I0
200.1.1.0/28	200.1.1.18	Fa0
0.0.0.0/0	150.214.141.26	I0

T.E. R2

Red	Próximo Salto	Interfaz
200.1.1.16/28	-	Fa0
200.1.1.0/29	-	Fa1
200.1.1.8/29	-	Fa2
0.0.0.0/0	200.1.1.17	Fa0

Febrero 2012 - P1 ap. e)

Apartado e) Si analizamos la dirección IP asignada al servidor web de Internet comprobamos que es la dirección 200.1.1.25. Esta dirección también se está usando para direccionar un dispositivo de la subred del PC D ya que se le había asignado el bloque 200.1.1.16/28. Por tanto, si el PC I intenta descargar una página web del servidor web de Internet en realidad todo el tráfico generado se dirigiría al dispositivo de esa subred configurado con la dirección IP 200.1.1.25.

El motivo de que ocurra esto es que el diseñador de la red ha usado un bloque público para crear la red privada en lugar de uno privado, que hubiera sido lo correcto. Esta decisión provoca que:

1. Si el tráfico va dirigido a los dispositivos de la red privada => todo funciona correctamente.
2. Si el tráfico va dirigido a cualquier red pública diferente de la utilizada (bloque 200.1.1.0/27) => todo funciona correctamente.
3. Si el tráfico va dirigido a la red pública utilizada (bloque 200.1.1.0/27) => los datagramas se reenvían hacia los dispositivos de la red de la empresa y no hacia los dispositivos de Internet direccionados con ese bloque público. Esto es un efecto colateral no deseable.

La solución es sencilla. El diseñador debe seleccionar un bloque privado y emplearlo para direccionar la red privada de la empresa.

Septiembre 2012 - P1 ap. e) y f)

Apartado e) A partir del enunciado y de los tiempos del diagrama:

$$d_{nodal\ router B} = d_{proc} + d_{cola} + d_{trans} + d_{prop} = 3,75\ \mu s + 0\ \mu s + 1\ \mu s + 0,5\ \mu s = 5,25\ \mu s$$

Apartado f) Tenemos varias alternativas: 1) Ethernet (R=10 Mbps), 2) Fast Ethernet (R=100 Mbps) y 3) Gigabit Ethernet (R = 1000 Mbps). Analizando los tiempos de transmisión el enlace de RA a RB es diez veces más lento que el enlace de RB a RC ya que el retardo de transmisión del primero es 10 µs y el del segundo 1 µs. Por tanto, las alternativas que se están utilizando en los enlaces quedan reducidas a estas dos hipótesis:

1. Enlace RA - RB = 10 Mbps y enlace de RB - RC = 100 Mbps.
2. Enlace RA - RB = 100 Mbps y enlace de RB - RC = 1000 Mbps.

Si calculamos la longitud del paquete (parámetro L) para la primera hipótesis y el primer enlace:

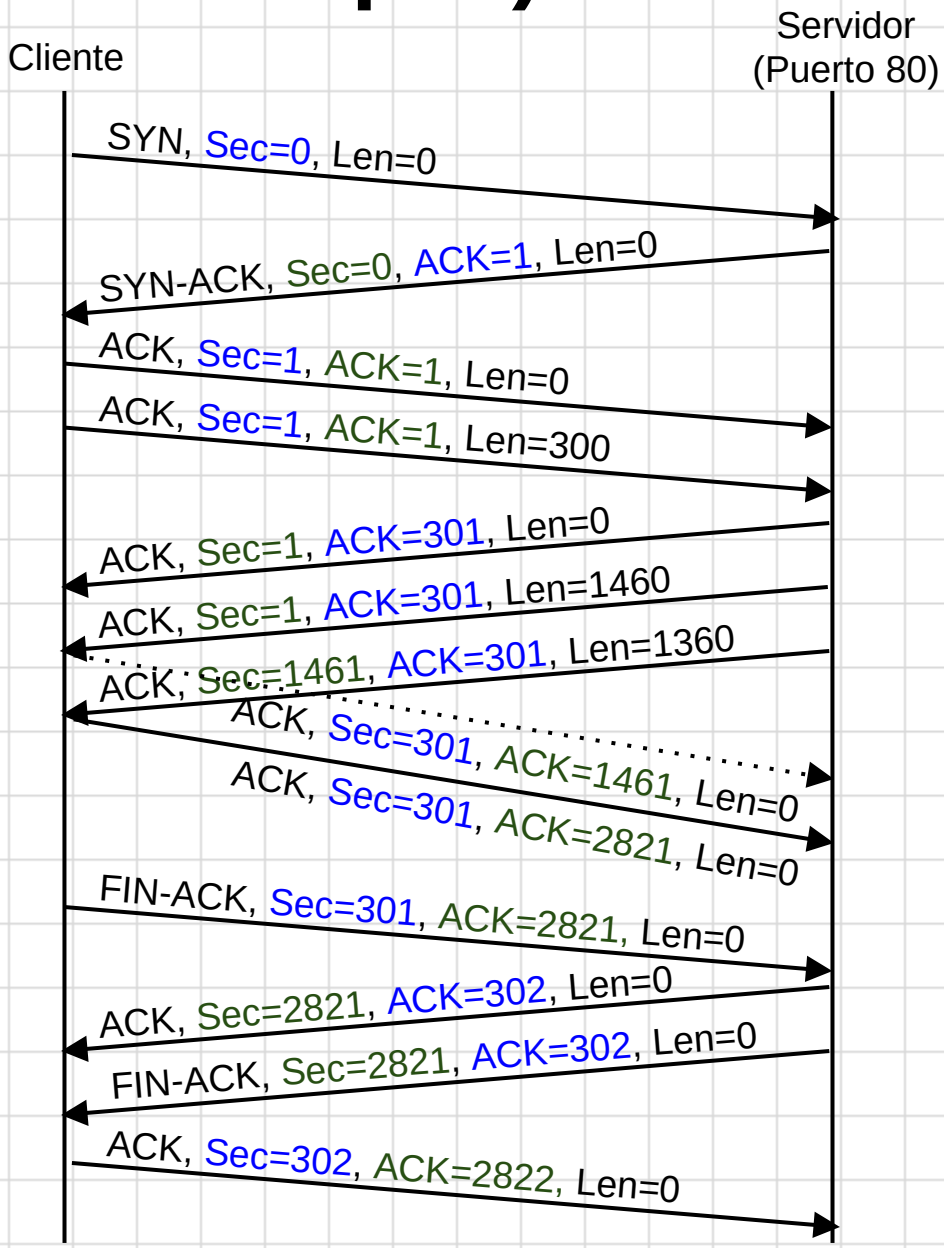
$$10\ \mu s = \frac{L}{10\ Mbps} \Longrightarrow L = 10 \times 10^{-6}\ s \times 10 \times 10^{6}\ bps = 100\ bits = 12,5\ bytes$$

Considerando que sólo la longitud de la E_PCI de Ethernet ocupa 26 bytes se puede concluir que no es posible que los enlaces tenga velocidades de 10 y 100 Mbps respectivamente.

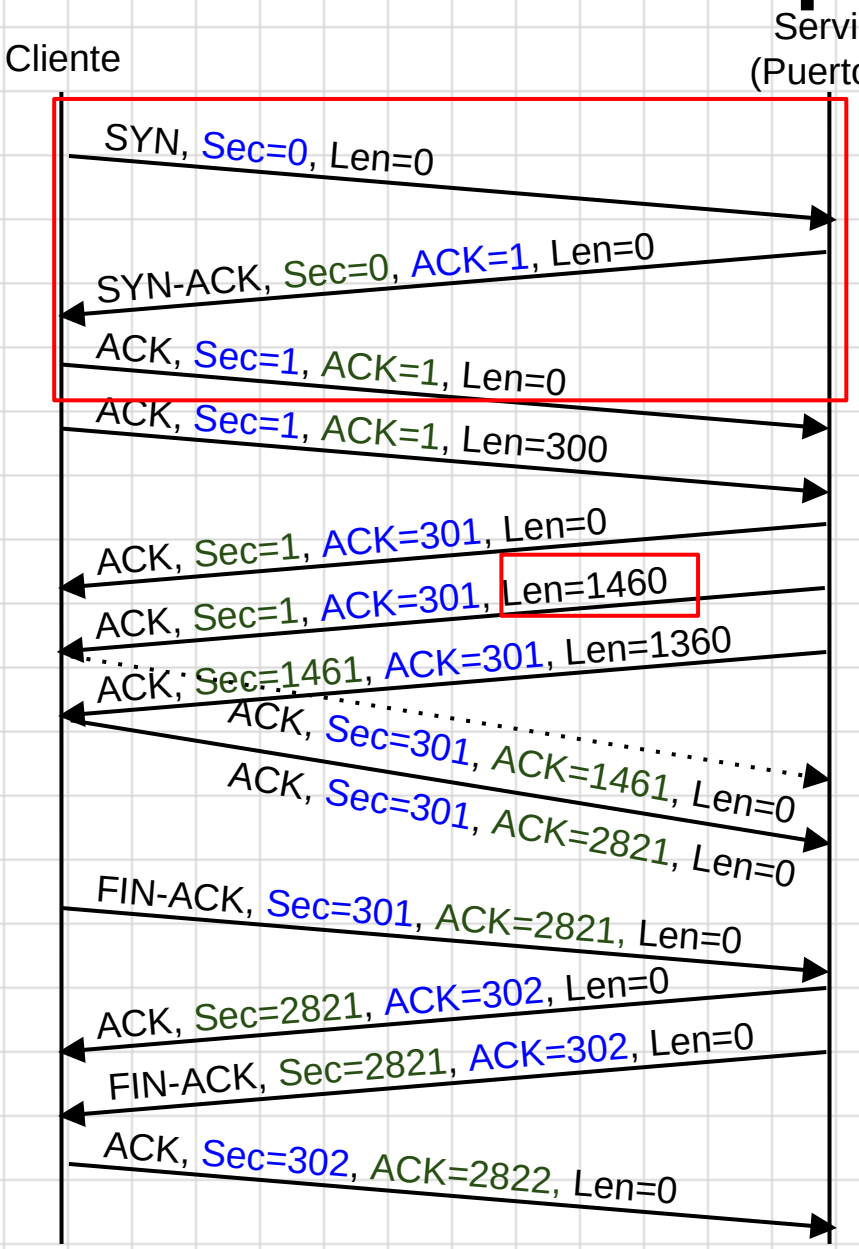
Realizando el mismo cálculo para la segunda hipótesis obtenemos un resultado de L = 125 bytes lo cual sí es posible ya que no entra en contradicción con ninguna norma estudiada. Por tanto, los tipos usados en los enlaces son los reflejados en la segunda hipótesis.

Curso 2012/13

Enero 2013 - P2 ap. a)



Enero 2013 - P2 ap. b), c) y d)

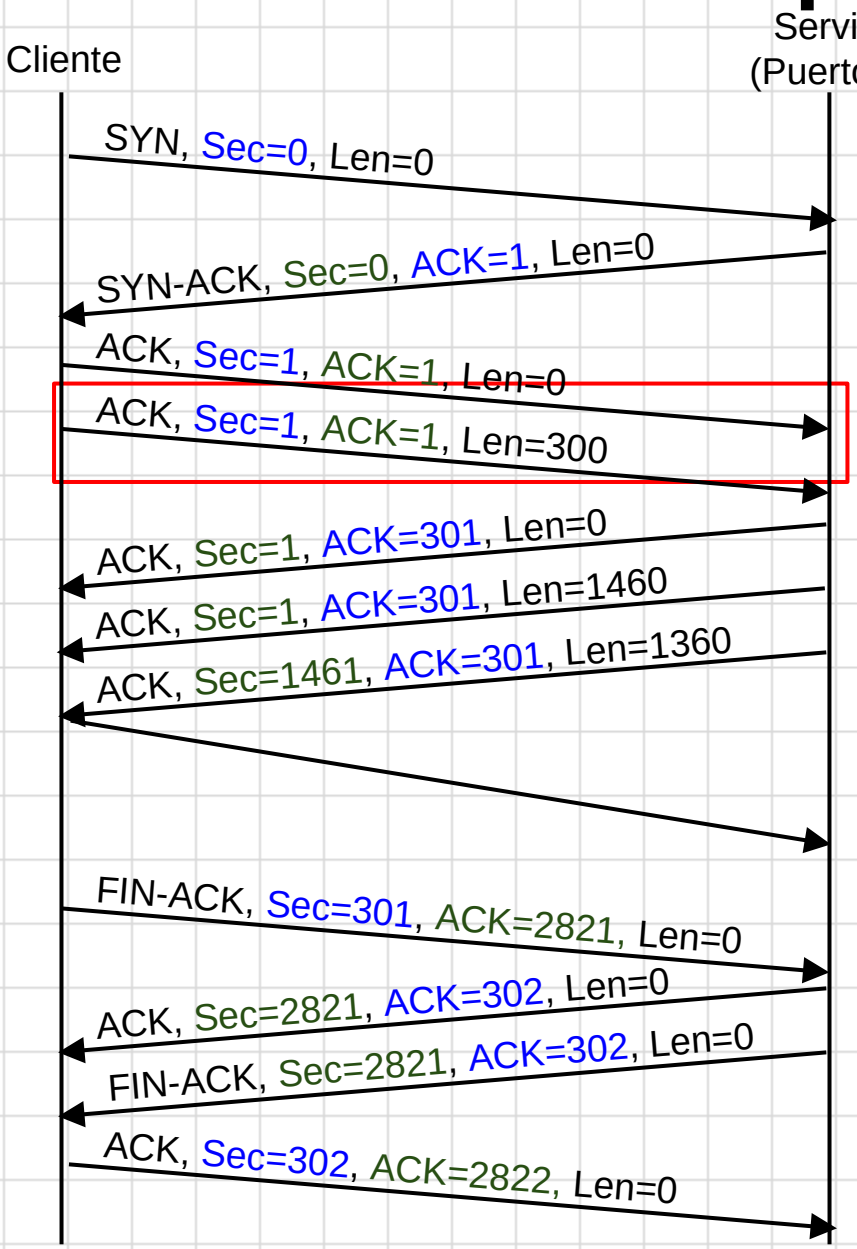


b) La pulsación sobre el enlace es inmediata por lo que en caso de ser persistente no habría dado tiempo a que la conexión se cerrara. Por tanto, al establecerse una nueva conexión se puede determinar que se ha usado HTTP no persistente.

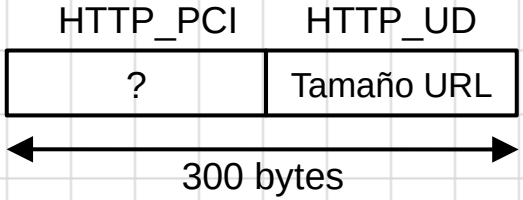
c) La TCP_SDU se ha tenido que segmentar. El primer segmento es de tamaño máximo por lo que el MSS es 1460 bytes.

d) Tal y como indica el enunciado se representan todas las T_PDU que se han intercambiado para descargar la página completa. Por tanto, esta página no tiene referencias a otros objetos ya que en ese caso tras finalizar la conexión se tendría que haber mostrado todo el proceso de descarga de los objetos referenciados.

Enero 2013 - P2 ap. e)

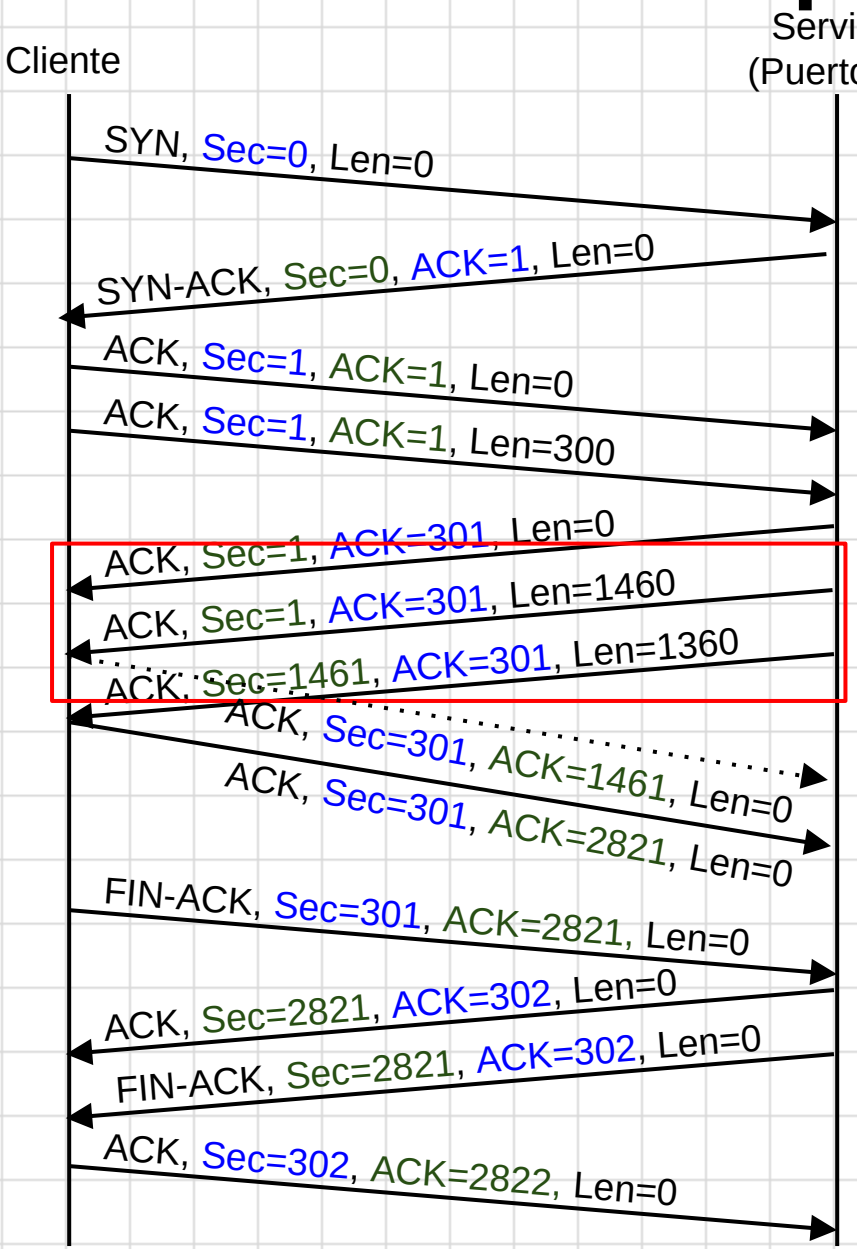


e) En la figura se representa el envío del GET de la página saludo.html. El tamaño de la HTTP_PDU es 300 bytes por lo que:

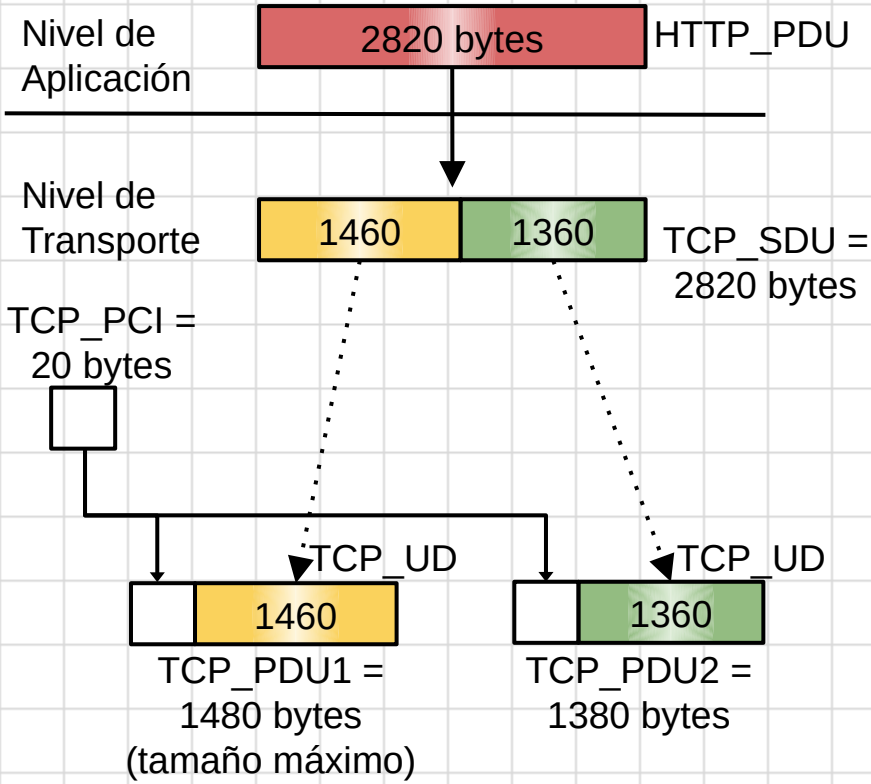


La URL: www.empresa.net/saludo.html tiene un tamaño de 27 bytes. Por tanto, el tamaño de la HTTP_UD = 27 bytes y el tamaño de la HTTP_PCI = 300 - 27 bytes = 273 bytes.

Enero 2013 - P2 ap. f)

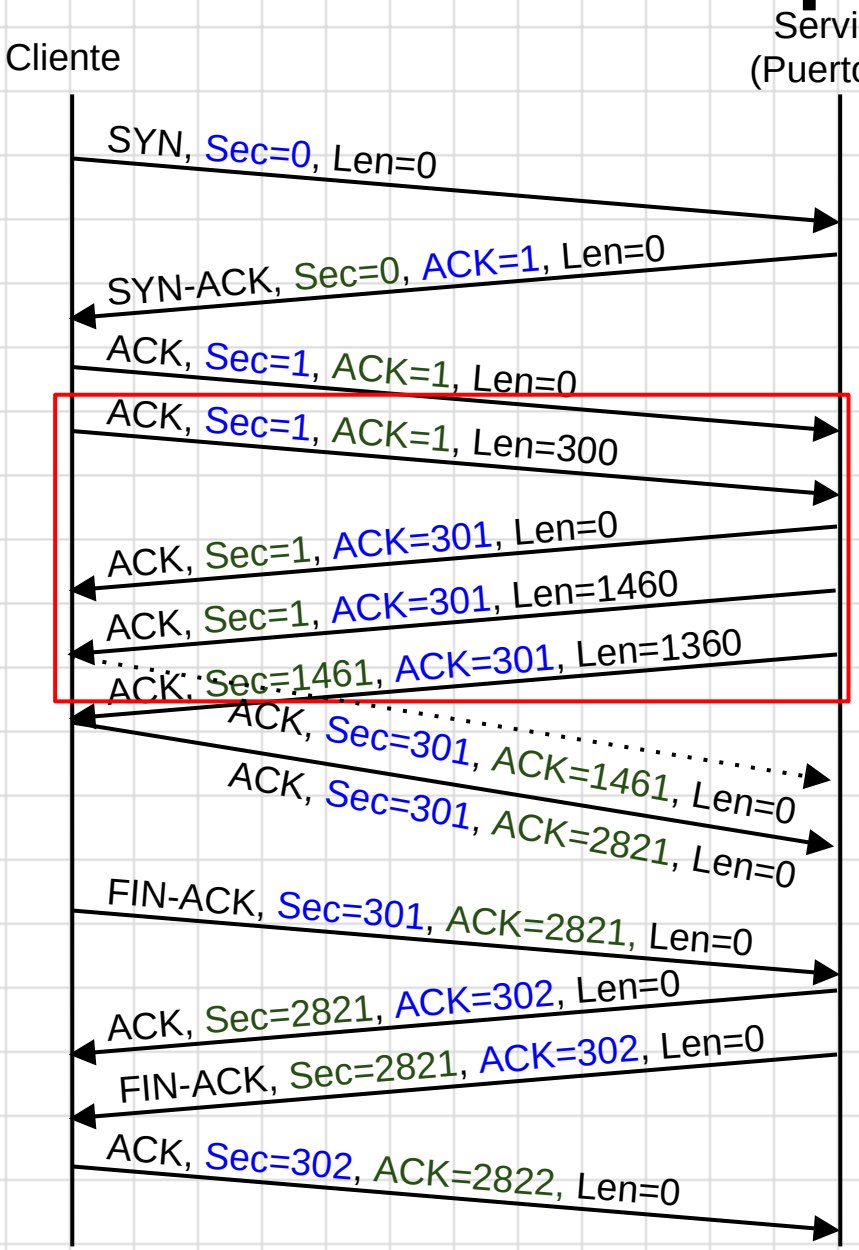


f) En la figura se observa que el servidor envía dos segmentos con datos (TCP_PDU). Esto indica que ha tenido que hacer segmentación.



El tamaño del objeto (HTTP_UD) se calcula a partir del tamaño de la HTTP_PDU (2820 bytes) y de la HTTP_PCI (273 bytes): 2820 - 273 bytes = 2547 bytes.

Enero 2013 - P2 ap. g)



g) El cliente ha solicitado la página web y el servidor ha enviado el objeto. Esto implica una de estas dos hipótesis:

1. El navegador no tenía el objeto en la caché, de ahí que el servidor lo haya enviado.

2. El navegador lo tenía en la caché. En este caso hizo un GET condicional pero el objeto se había modificado, por lo que el servidor envió la versión actualizada.

Septiembre 2013 - P1 ap. a) y b)

Apartado a) Tenemos que calcular la longitud del enlace que corresponde al parámetro *d*. De este parámetro depende el tiempo de propagación que se puede extraer de la figura (d_{prop} = 2 µs). Además, en el enunciado se proporciona la velocidad de propagación s = 2 x 10^8 m/s. Por tanto, la longitud del enlace puede calcularse como:

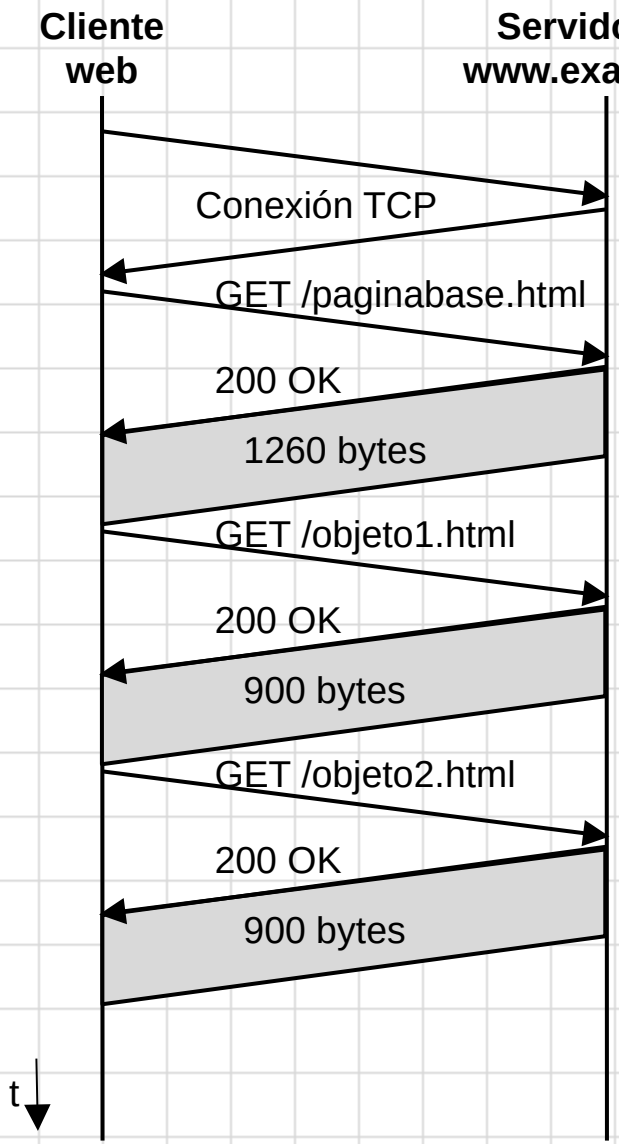
$$d_{prop} = \frac{d}{2 \times 10^8\ m/s} = 2 \times 10^{-6}\ s \Longrightarrow d = 2 \times 10^8\ m/s \times 2 \times 10^{-6}\ s = 400\ m$$

Apartado b) El retardo de cola considerando las condiciones del problema se puede calcular como:

$$d_{cola}(MAC\_PDU2) = d_{trans}(MAC\_PDU1) = 6\ \mu s$$

Curso 2013/14

Enero 2014 - P1 ap. a)



HTTP_PDU enviadas por el cliente:

Tipo: Petición (GET). Cantidad: 3 (ver diagrama).

Tamaños:

1. GET /paginabase.html:

HTTP_PCI	HTTP_UD
240 bytes	URL

URL: www.examen.net/paginabase.html (Tamaño: 30 bytes)

Tamaño de la primera HTTP_PDU = 240 + 30 = 270 bytes.

2. GET /objeto1.html:

HTTP_PCI	HTTP_UD
240 bytes	27 bytes

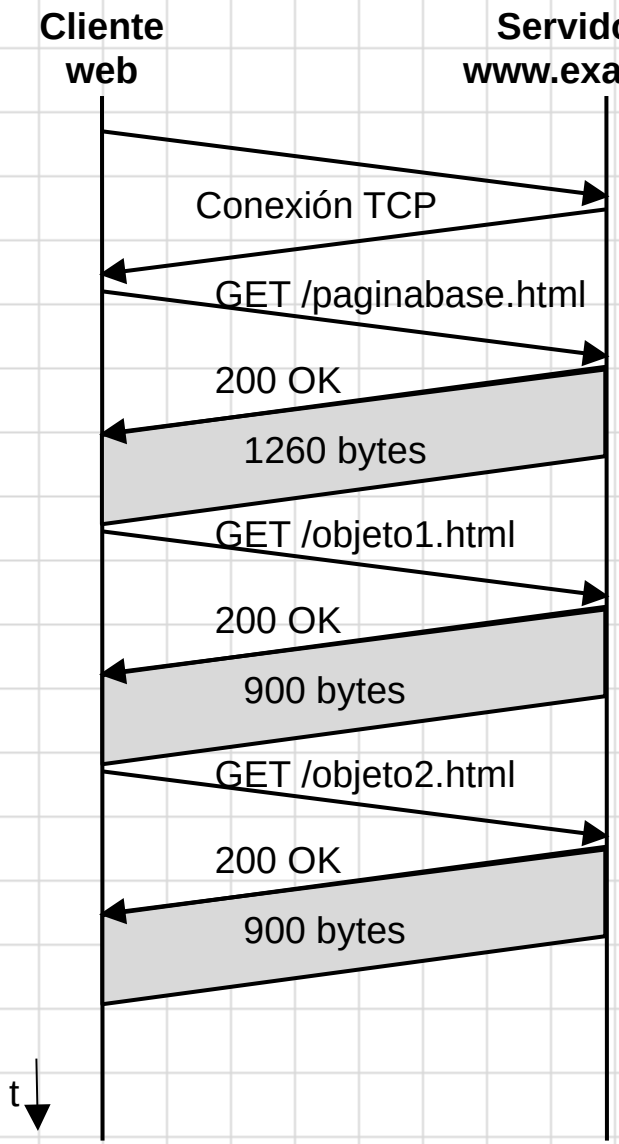
Tamaño de la segunda HTTP_PDU = 267 bytes.

3. GET /objeto2.html:

HTTP_PCI	HTTP_UD
240 bytes	27 bytes

Tamaño de la tercera HTTP_PDU = 267 bytes.

Enero 2014 - P1 ap. a)



HTTP_PDU enviadas por el servidor:

Tipo: Respuesta (200 OK). Cantidad: 3 (ver diagrama).

Tamaños:

1. 200 OK (paginabase.html):

HTTP_PCI	HTTP_UD
240 bytes	Objeto

Tamaño del objeto paginabase.html = 1260 bytes.

Tamaño de la primera HTTP_PDU = 240 + 1260 = 1500 bytes.

2. 200 OK (objeto1.html):

HTTP_PCI	HTTP_UD
240 bytes	900 bytes

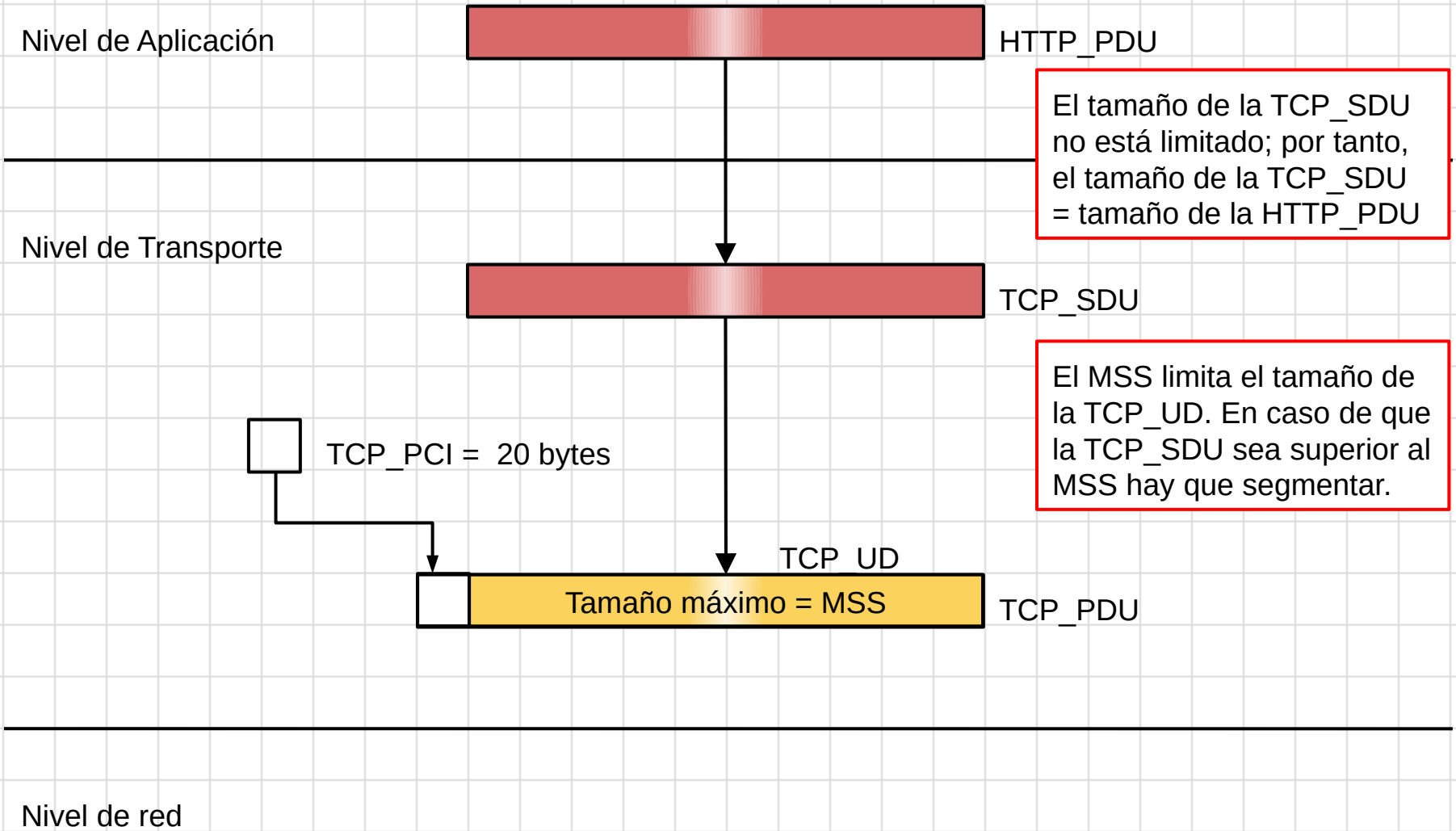
Tamaño de la segunda HTTP_PDU = 1140 bytes.

3. 200 OK (objeto2.html):

HTTP_PCI	HTTP_UD
240 bytes	900 bytes

Tamaño de la tercera HTTP_PDU = 1140 bytes.

Enero 2014 - P1 ap. b)



Enero 2014 - P1 ap. b)

Para los tamaños de las HTTP_PDU enviadas por el cliente:

HTTP_PDU1 = TCP_SDU1 = 270 bytes,

HTTP_PDU2 = TCP_SDU2 = 267 bytes,

HTTP_PDU3 = TCP_SDU3 = 267 bytes,

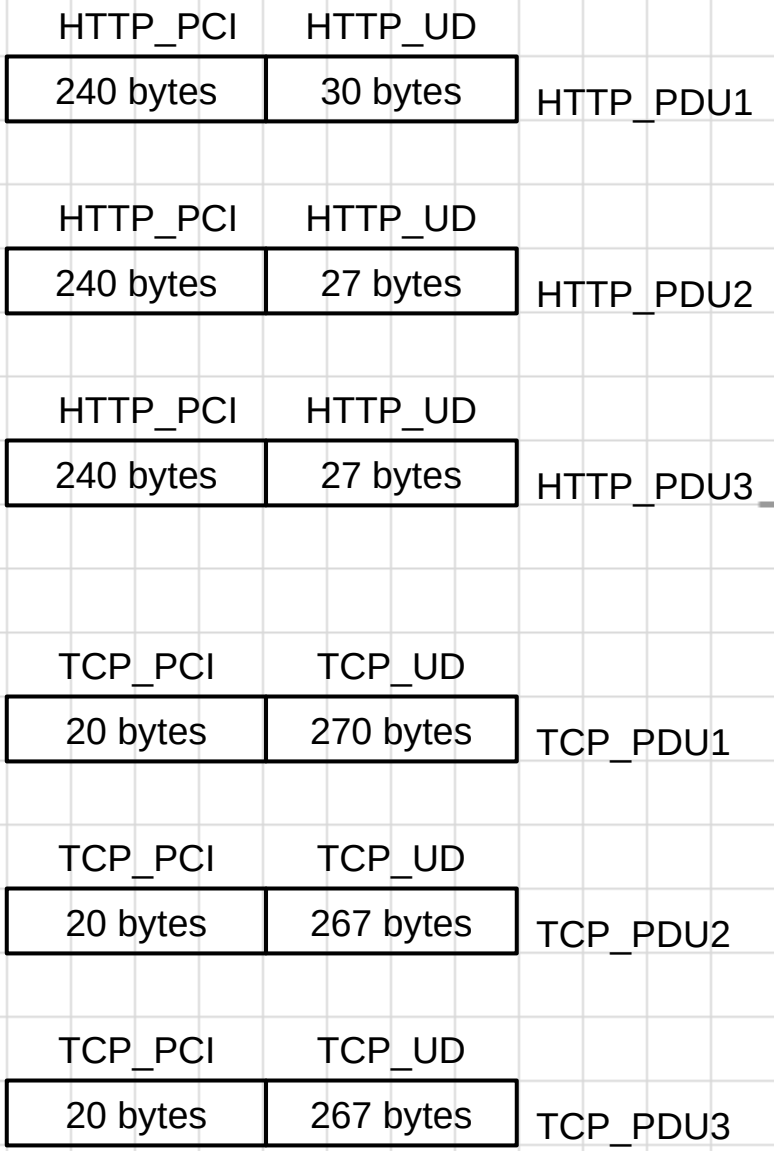
se observa que en ningún caso la TCP_SDU supera el MSS (1260 bytes) por lo que no hay que segmentar.

Una vez encapsuladas las HTTP_PDU el tamaño de los segmentos (TCP_PDU) es el siguiente:

TCP_PDU1 = 20 + 270 = 290 bytes,

TCP_PDU2 = 20 + 267 = 287 bytes,

TCP_PDU3 = 20 + 267 = 287 bytes.



Enero 2014 - P1 ap. b)

Para los tamaños de las HTTP_PDU enviadas por el servidor:

HTTP_PDU1 = TCP_SDU1 = 1500 bytes,

HTTP_PDU2 = TCP_SDU2 = 1140 bytes,

HTTP_PDU3 = TCP_SDU3 = 1140 bytes,

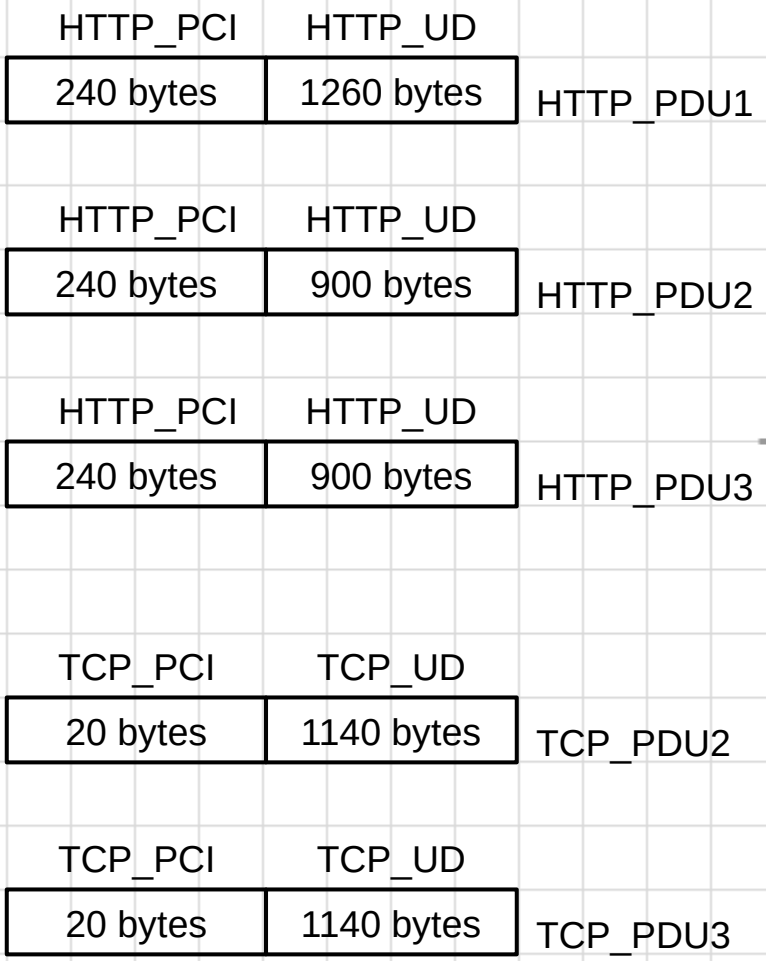
se observa que en el caso de la primera respuesta la TCP_SDU supera el MSS (1260 bytes) por lo que hay que segmentar. La TCP_SDU de las respuestas 2 y 3 no supera el MSS por lo que no hay que segmentar en esos casos.

Una vez encapsuladas las HTTP_PDU 2 y 3 el tamaño de los segmentos (TCP_PDU) es el siguiente:

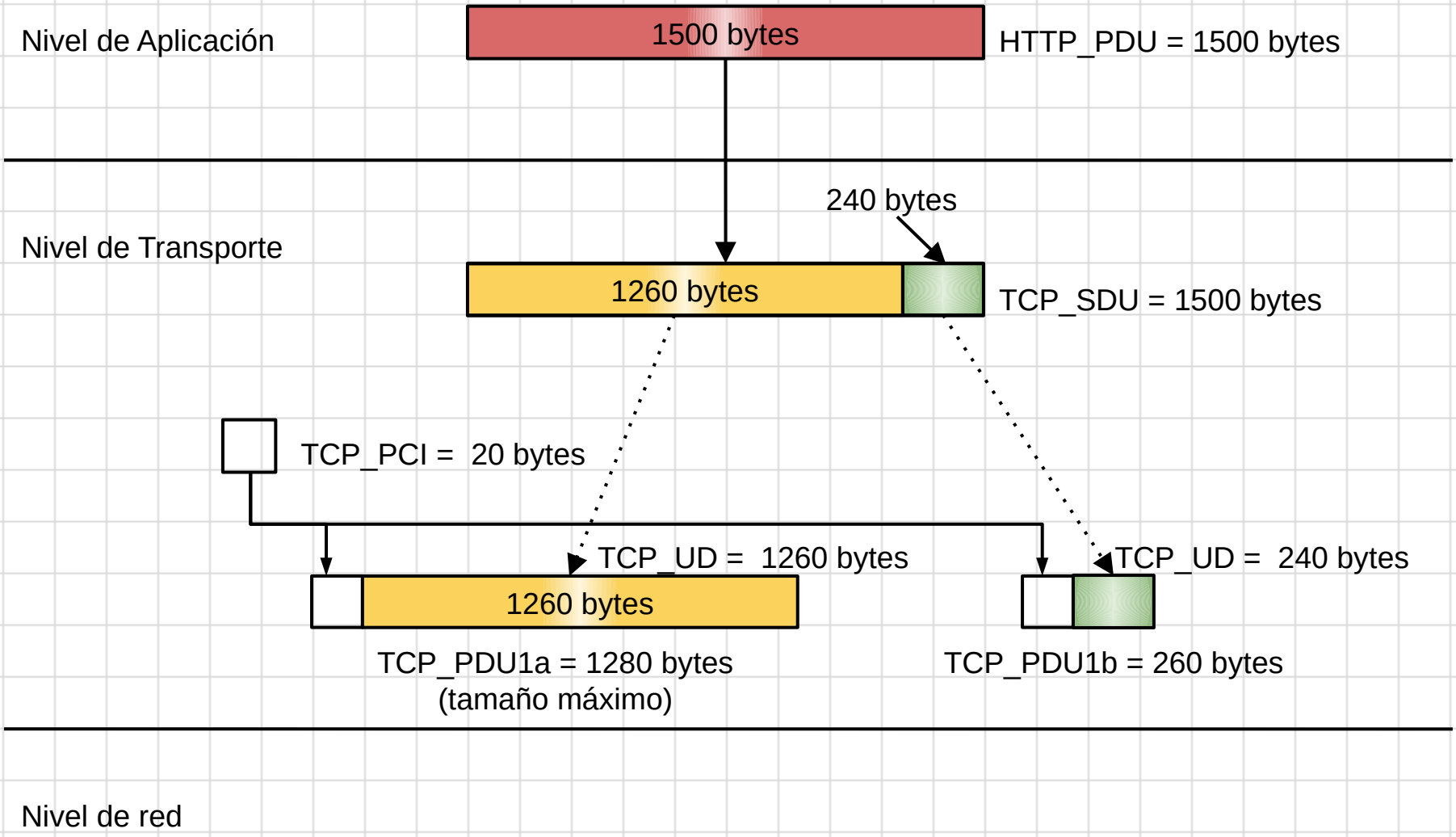
TCP_PDU2 = 20 + 1140 = 1160 bytes,

TCP_PDU3 = 20 + 1140 = 1160 bytes.

Queda hacer la segmentación de la primera respuesta.

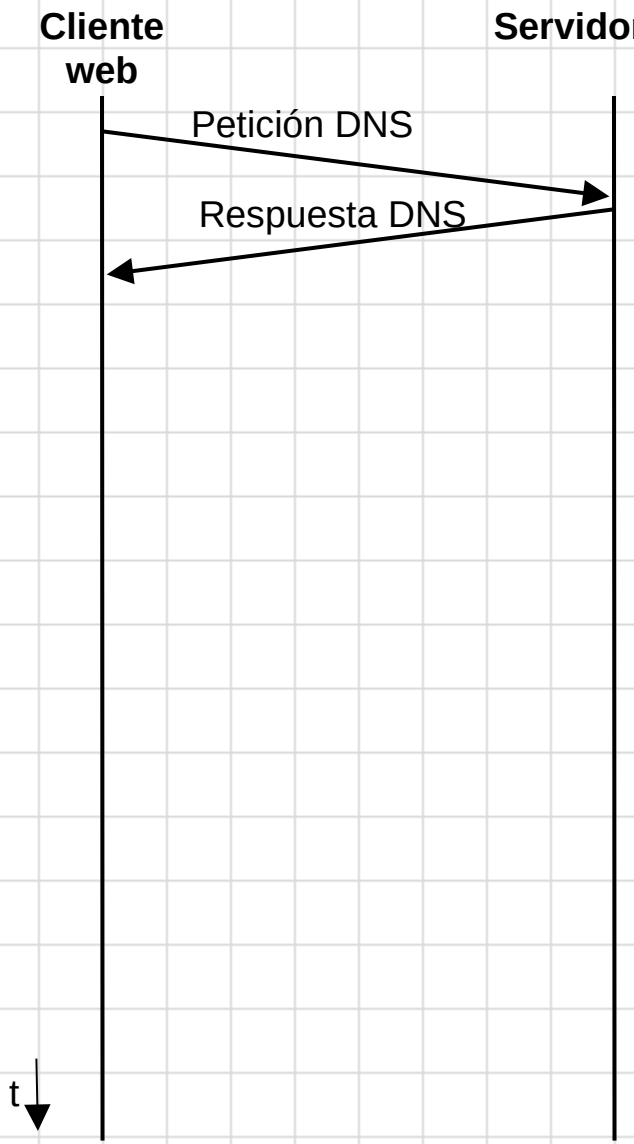


Enero 2014 - P1 ap. b)



Curso 2015/16

Enero 2016 - P1 ap. a)



Antes de crear la conexión TCP necesitamos resolver el nombre del servidor web: www.informacion.lab

El PC envía una petición DNS solicitando al servidor DNS que resuelva el nombre. En el enunciado indica que el tamaño de la DNS_PDU es de 200 bytes.

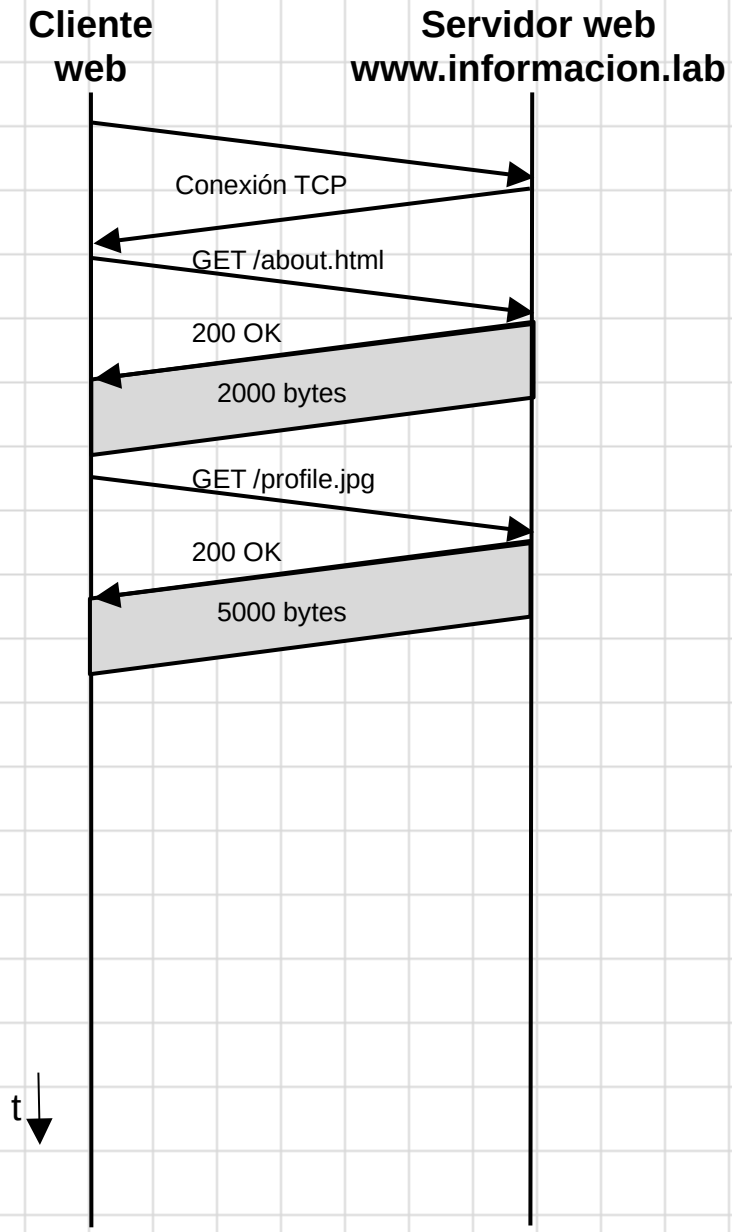
Pet. DNS	DNS_PDU = 200 bytes

El PC recibe una respuesta DNS del servidor DNS que resuelve el nombre solicitado.

Resp. DNS	DNS_PDU = 200 bytes

Una vez resuelto el nombre ya se puede establecer la conexión TCP con el servidor web.

Enero 2016 - P1 ap. a)



Solicitamos la página base y el objeto referenciado. Como la conexión es persistente no hace falta cerrar y abrir una segunda conexión.

1. GET /about.html:

HTTP_PCI	HTTP_UD
300 bytes	30 bytes

Tamaño de la 1ª petición: HTTP_PDU = 300 + 30 = 330 bytes.

2. 200 OK (about.html):

HTTP_PCI	HTTP_UD
300 bytes	2000 bytes

Tamaño de la 1ª respuesta: HTTP_PDU = 2300 bytes.

3. GET /profile.jpg:

HTTP_PCI	HTTP_UD
300 bytes	31 bytes

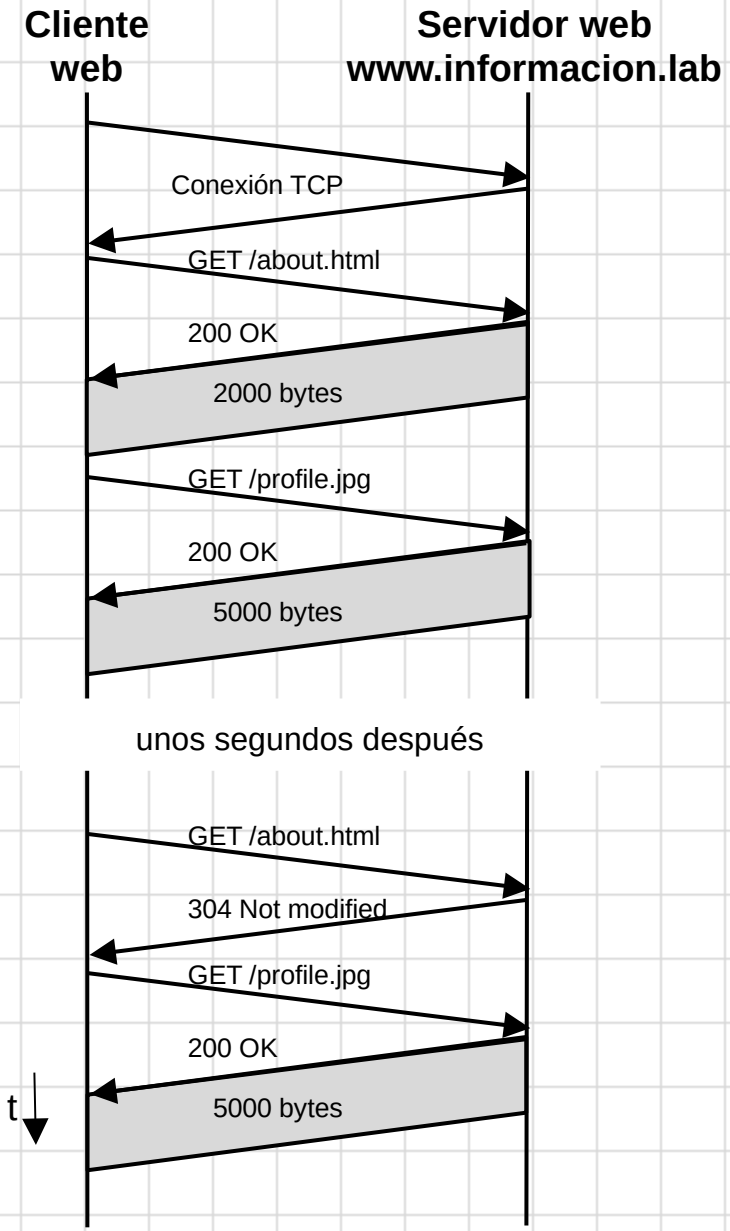
Tamaño de la 2ª petición: HTTP_PDU = 331 bytes.

4. 200 OK (profile.jpg):

HTTP_PCI	HTTP_UD
300 bytes	5000 bytes

Tamaño de la 2ª respuesta: HTTP_PDU = 5300 bytes.

Enero 2016 - P1 ap. a)



Pasados unos segundos se recarga la página. Como el tiempo entre la carga y la recarga es menor que el timeout de HTTP la conexión TCP aún no se ha cerrado. Al tener los objetos en la caché de páginas los GET son condicionales.

1. GET cond. /about.html:

HTTP_PCI	HTTP_UD
300 bytes	30 bytes

Tamaño de la 1ª petición: HTTP_PDU = 300 + 30 = 330 bytes.

(Nota: en realidad la HTTP_PCI aumentaría un poco su tamaño ya que se incluyen nuevas líneas de cabecera)

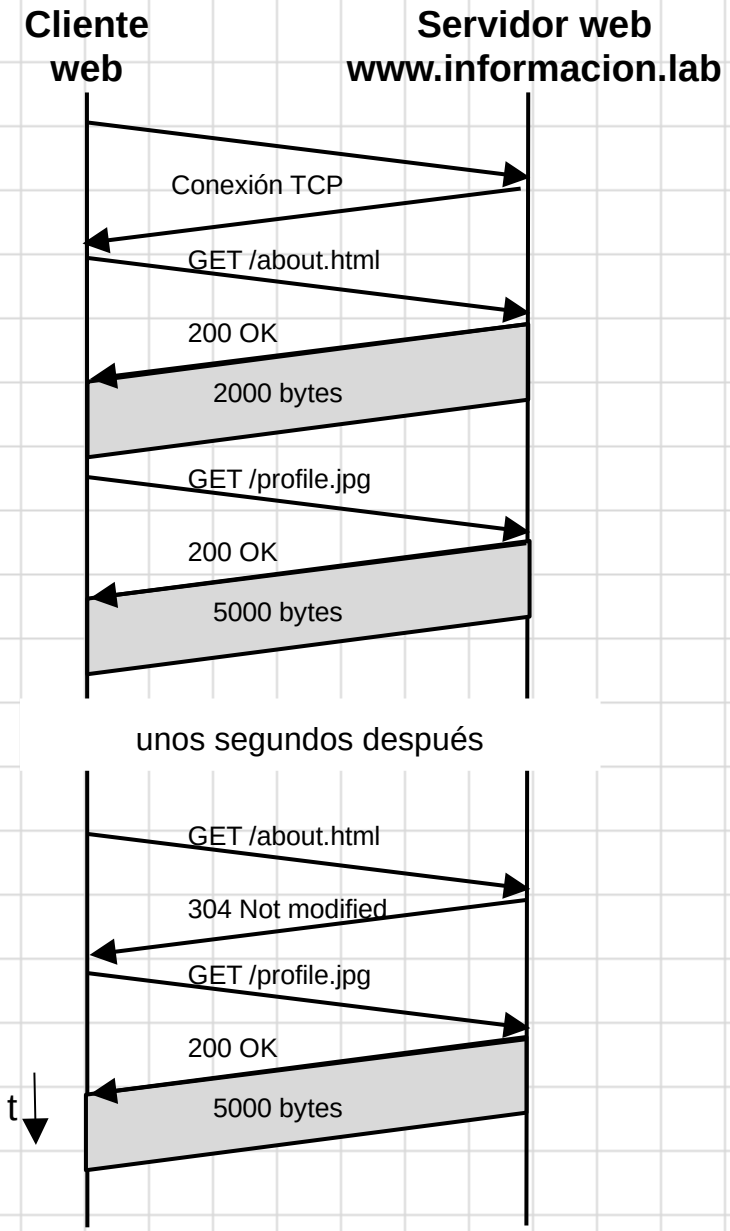
2. 304 Not modified (about.html):

HTTP_PCI	HTTP_UD
300 bytes	0 bytes

Tamaño de la 1ª respuesta: HTTP_PDU = 300 bytes.

Como el GET es condicional y no se ha modificado el objeto se devuelve la línea de estado “304 Not modified” y no se envía el objeto.

Enero 2016 - P1 ap. a)



Se solicita el objeto referenciado. En el enunciado se comenta que ha cambiado por lo que el servidor enviará la nueva versión de este objeto.

3. GET cond. /profile.jpg:

HTTP_PCI	HTTP_UD
300 bytes	31 bytes

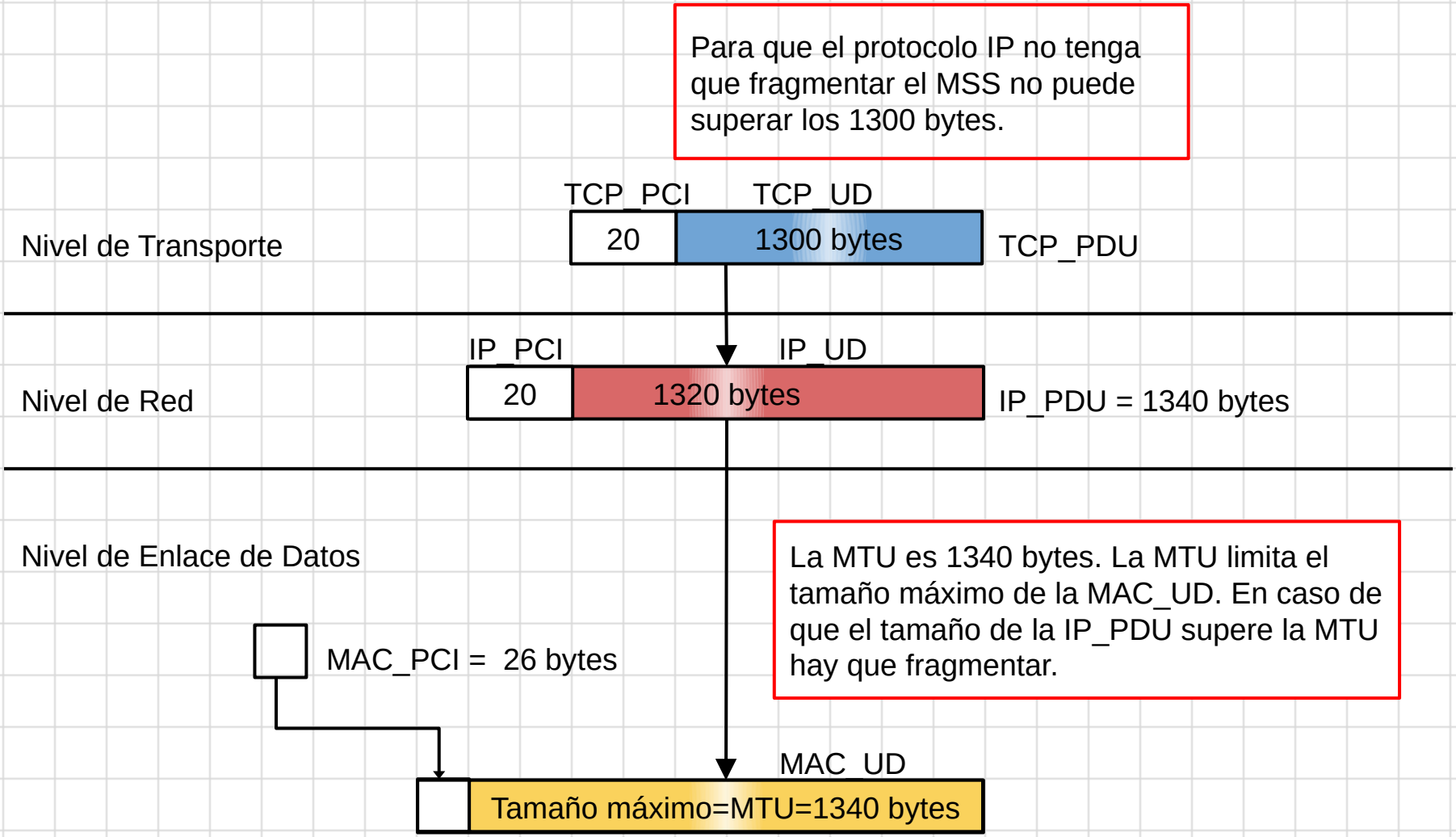
Tamaño de la 2ª petición: HTTP_PDU = 331 bytes.

4. 200 OK (profile.jpg):

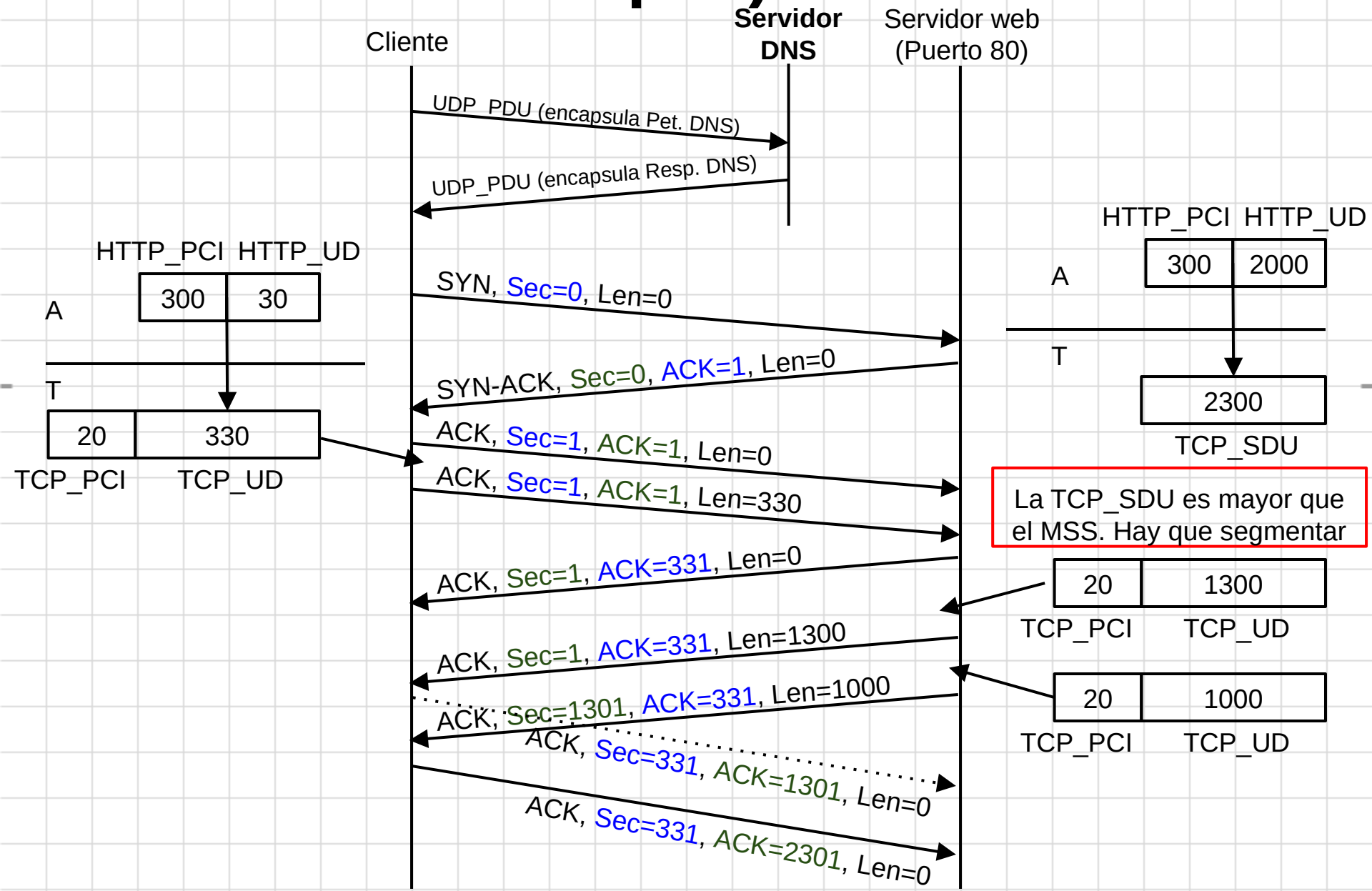
HTTP_PCI	HTTP_UD
300 bytes	5000 bytes

Tamaño de la 2ª respuesta: HTTP_PDU = 5300 bytes.

Enero 2016 - P1 ap. b)

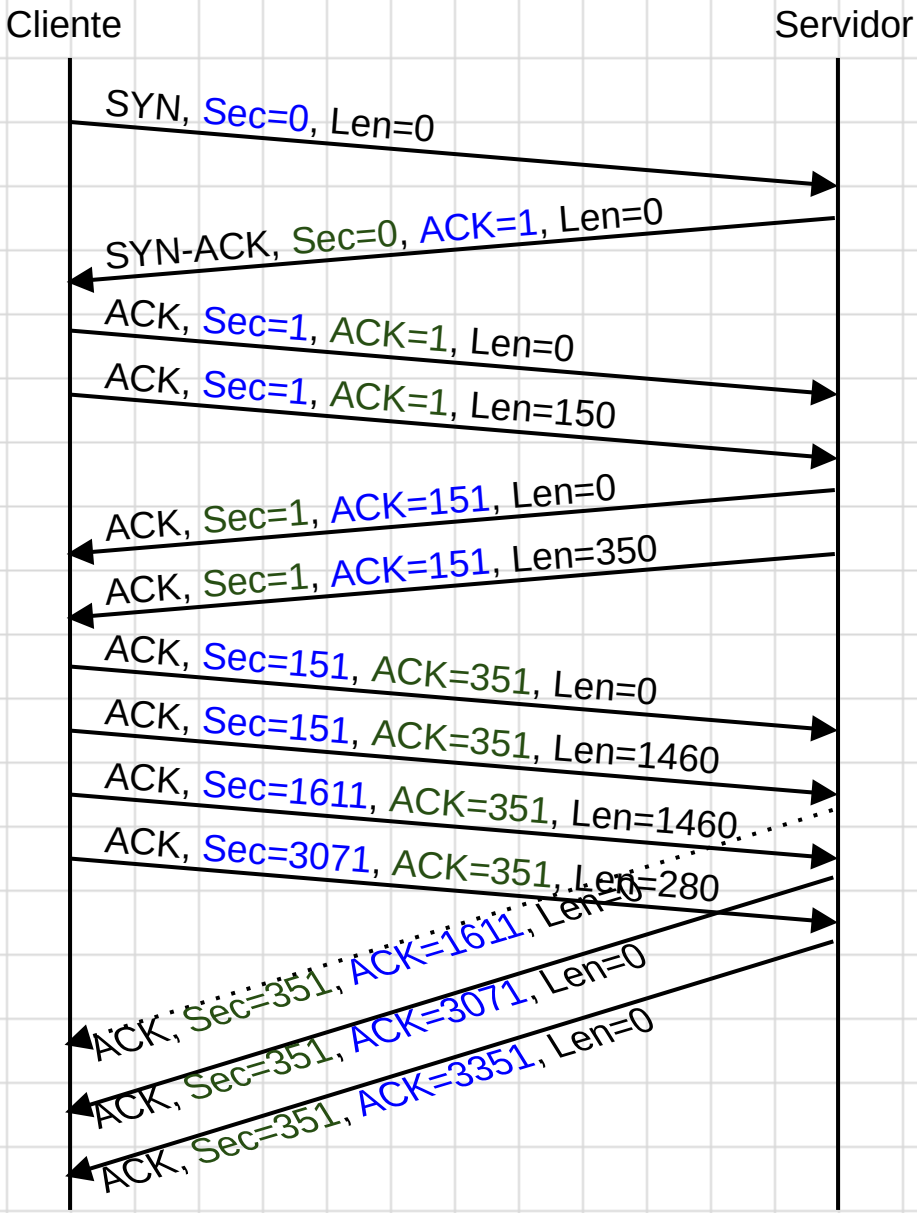


Enero 2016 - P1 ap. b)



Curso 2016/17

Enero 2017 - Apartado 5

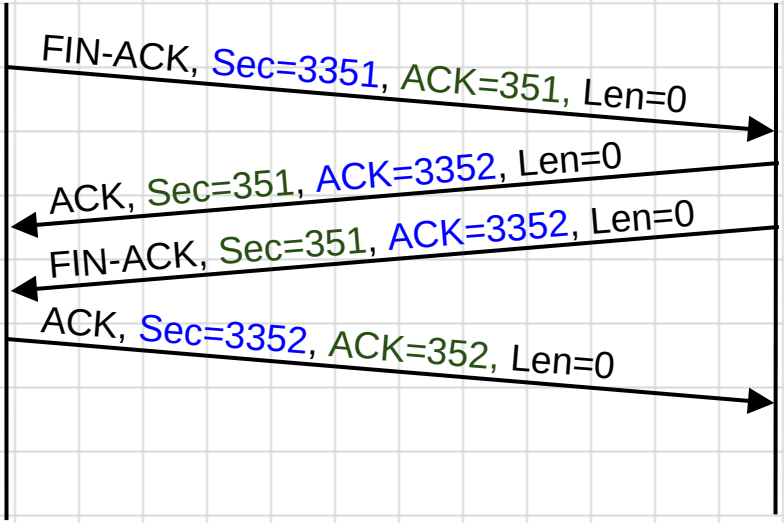


El fichero ocupa 3200 bytes por lo que hay que segmentar.

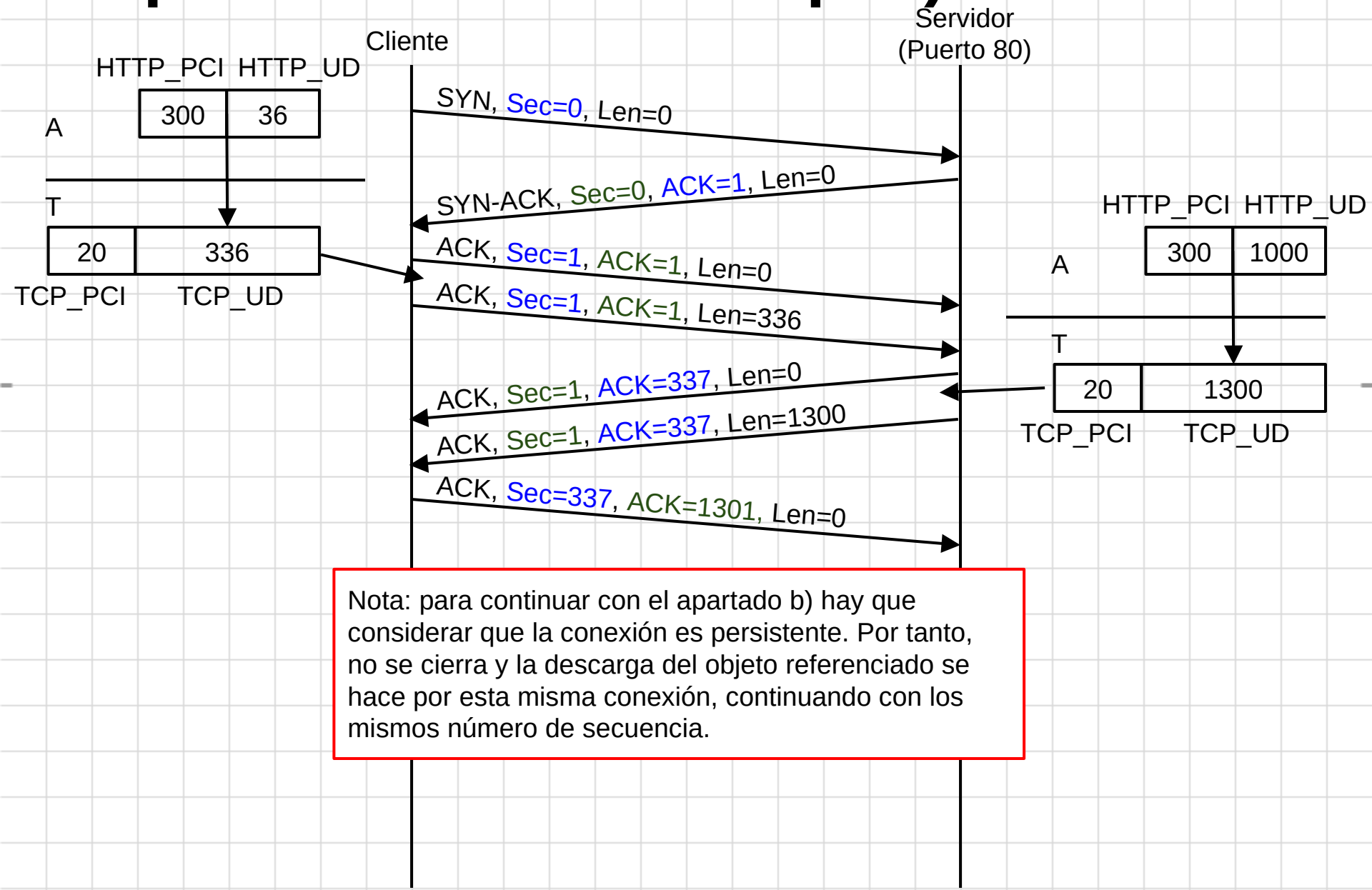
El fin de conexión se representa en la siguiente transparencia.

Enero 2017 - Apartado 5 (fin de conexión)

Suponiendo que la inicia el cliente.

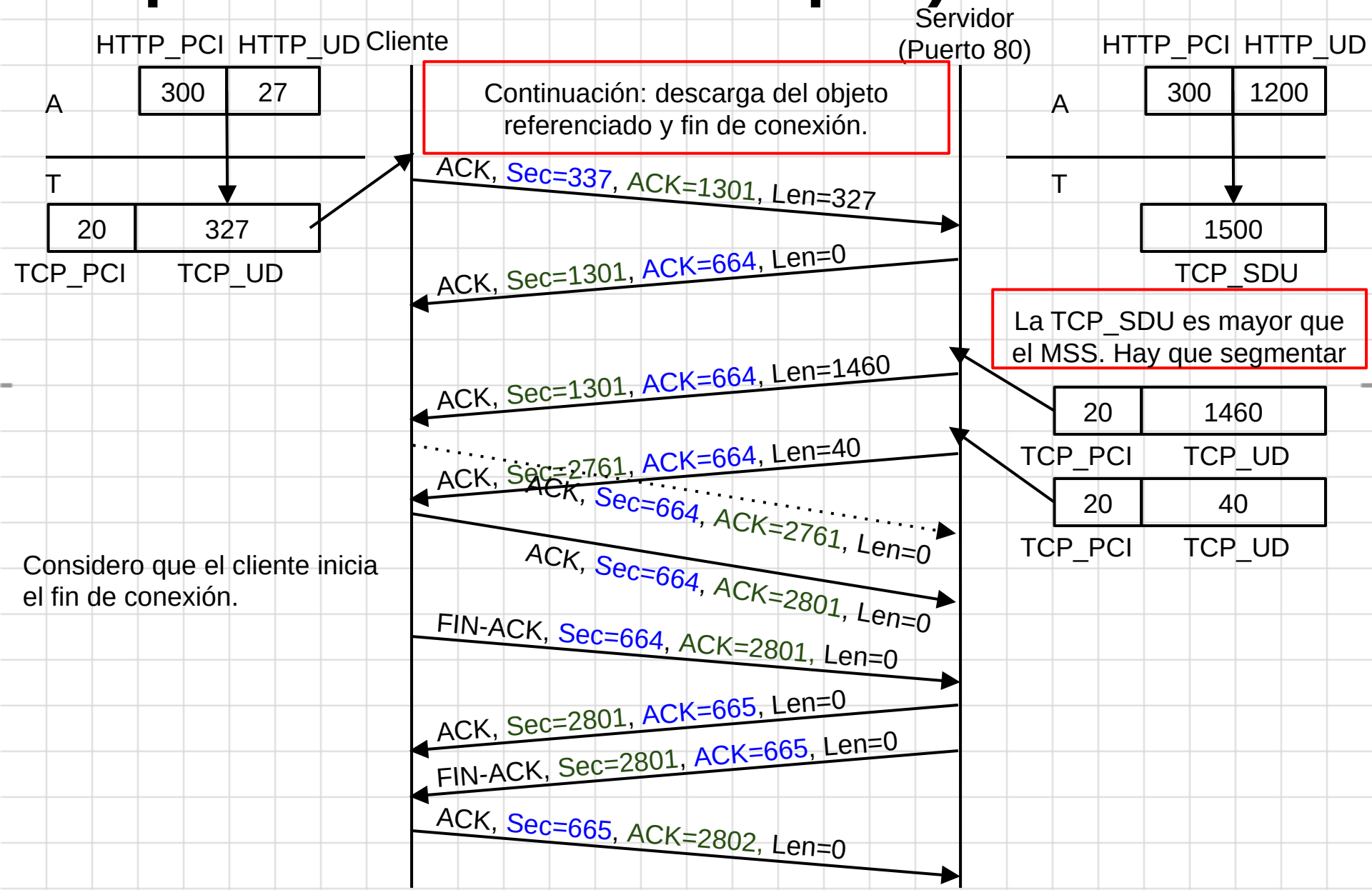


Septiembre 2017 - P2 ap. a)



Nota: para continuar con el apartado b) hay que considerar que la conexión es persistente. Por tanto, no se cierra y la descarga del objeto referenciado se hace por esta misma conexión, continuando con los mismos número de secuencia.

Septiembre 2017 - P2 ap. b)



Considero que el cliente inicia el fin de conexión.

Diciembre 2017- P1

En primer lugar vamos a determinar cuántas direcciones IP necesitamos en cada subred:

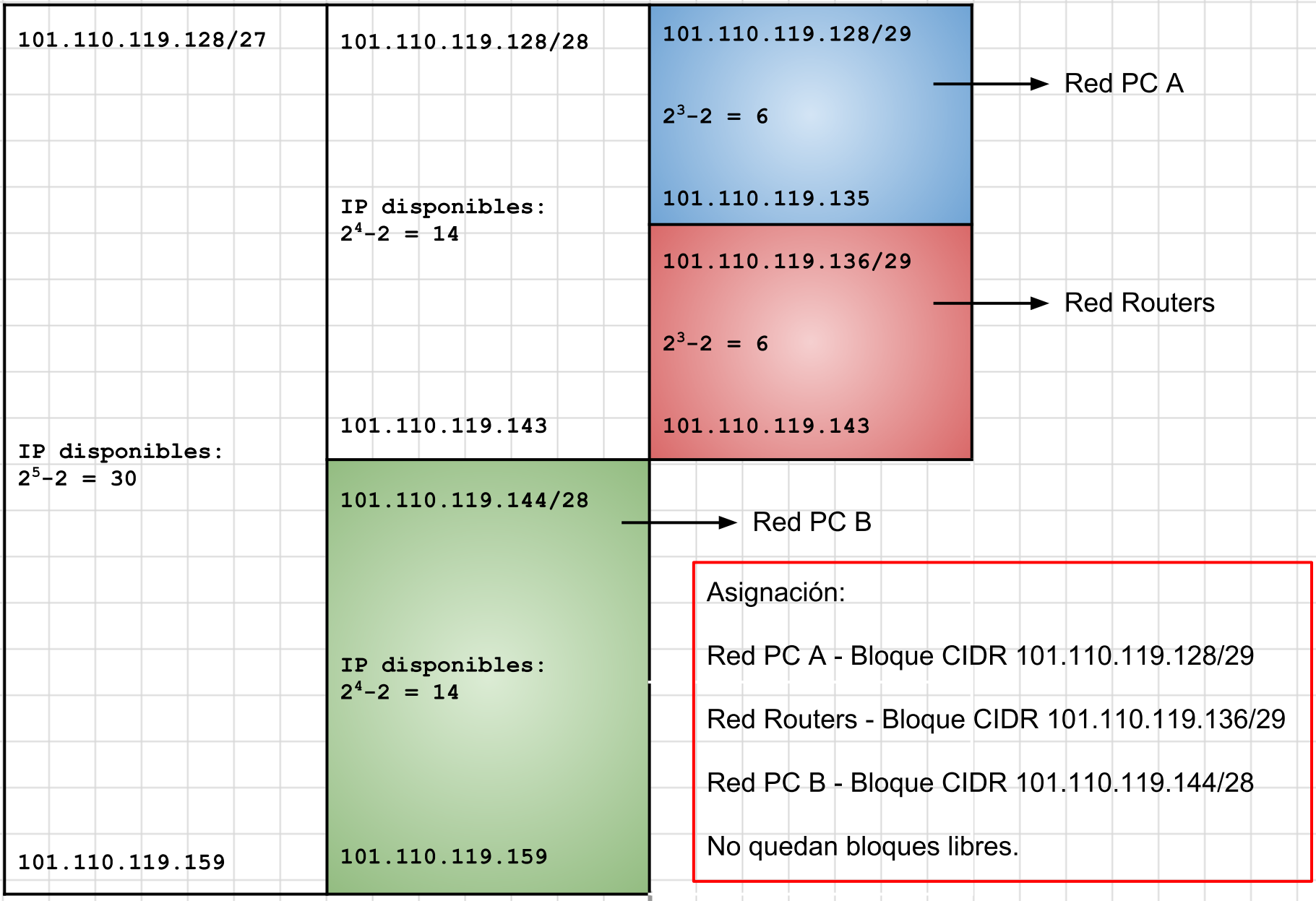
- **Subred PC A**: hay 2 sistemas finales y la interfaz Fa0 del router R3. Necesitamos **3 direcciones IP**.
- **Subred Routers**: hay 3 interfaces de routers: Fa1 de R3, Fa1 de R2 y Fa0 de R1. Por tanto, necesitamos **3 direcciones IP**.
- **Subred PC B**: hay 4 sistemas finales (más 6 que se conectarán al hub1) y la interfaz Fa0 de R2. Necesitamos **11 direcciones IP**.

A continuación, vamos a calcular los prefijos de red considerando que hay que dejar sin asignar el mayor número de IP para futuras ampliaciones de la red:

- **Subred PC A**: con un prefijo /29 dedicamos 29 bits para la red y 3 bits para la parte de host. Con 3 bits se pueden direccionar 2^3 - 2 = 6 dispositivos con nivel de red (sistemas finales y routers) por lo que tenemos suficientes direcciones. Necesitamos asignar por tanto un prefijo /29.
- **Subred Routers**: es similar a la subred PC A. Necesitamos un prefijo /29.
- **Subred PC B**: con un prefijo /29 no tendríamos suficientes (ver asignación de la subred PC A). Necesitamos un /28. Con un prefijo /28 dedicamos 28 bits para la red y 4 bits para la parte de host. Con 4 bits se pueden direccionar 2^4 - 2 = 14 dispositivos con nivel de red (sistemas finales y routers).

La asignación final de prefijos se muestra en la tabla de la siguiente transparencia.

Diciembre 2017- P1



Asignación:

Red PC A - Bloque CIDR 101.110.119.128/29

Red Routers - Bloque CIDR 101.110.119.136/29

Red PC B - Bloque CIDR 101.110.119.144/28

No quedan bloques libres.

Diciembre 2017- P1 y P2

1) Se pide proporcionar el identificador de red, la máscara de subred y la dirección de broadcast de cada una de las subredes:

Subred	Identificador	Máscara	D. Broadcast
Red PC A	101.110.119.128	255.255.255.248	101.110.119.135
Red Routers	101.110.119.136	255.255.255.248	101.110.119.143
Red PC B	101.110.119.144	255.255.255.240	101.110.119.159

2) No necesario implementar NAT ya que el ISP nos ha proporcionado un bloque de direcciones IP públicas, es decir, el bloque CIDR utilizado no se encuentra dentro de uno de los tres rangos de IP privadas existente.

Diciembre 2017- P3

Las direcciones IP de las interfaces de los routers de acuerdo a la asignación de subredes realizada sería:

Router	Interfaz	Dirección IP
R1	Fa0	101.110.119.137
	Fa1	120.60.30.1
R2	Fa0	101.110.119.145
	Fa1	101.110.119.138
R3	Fa0	101.110.119.129
	Fa1	101.110.119.139
Rext	Fa0	120.60.30.2
	Fa1	No se dispone de información para conocerla

Diciembre 2017- P3

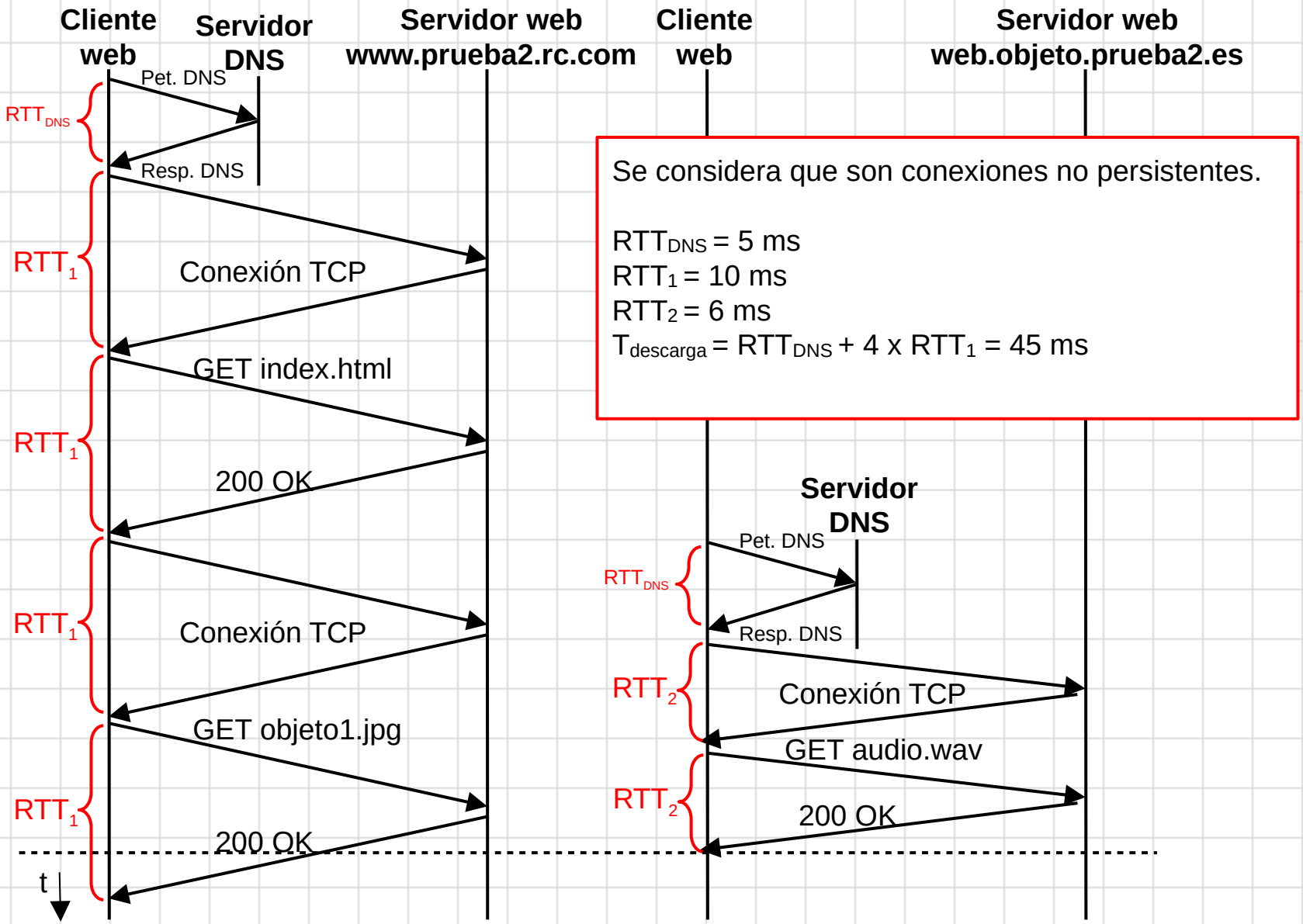
La tabla de enrutamiento de R1 quedaría como se muestra a continuación:

T.E. R1

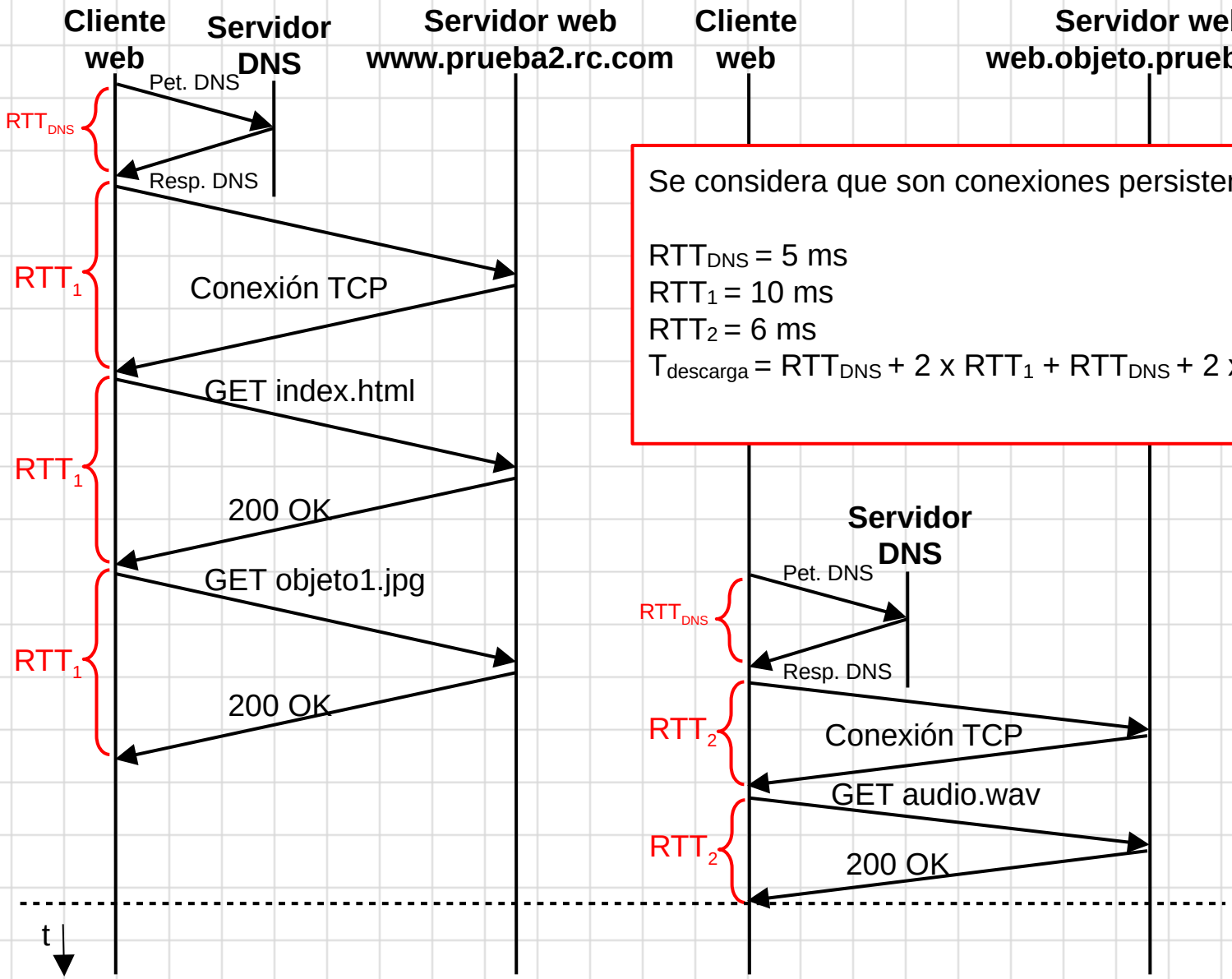
Red	Próximo Salto	Interfaz
101.110.119.136/29	-	Fa0 (101.110.119.137)
120.60.30.0/30	-	Fa1 (120.60.30.1)
101.110.119.128/29	101.110.119.139	Fa0 (101.110.119.137)
101.110.119.144/28	101.110.119.138	Fa0 (101.110.119.137)
0.0.0.0/0	120.60.30.2	Fa1 (120.60.30.1)

Curso 2018/19

Septiembre 2019 - P2, ap. a)



Septiembre 2019 - P2, ap. b)

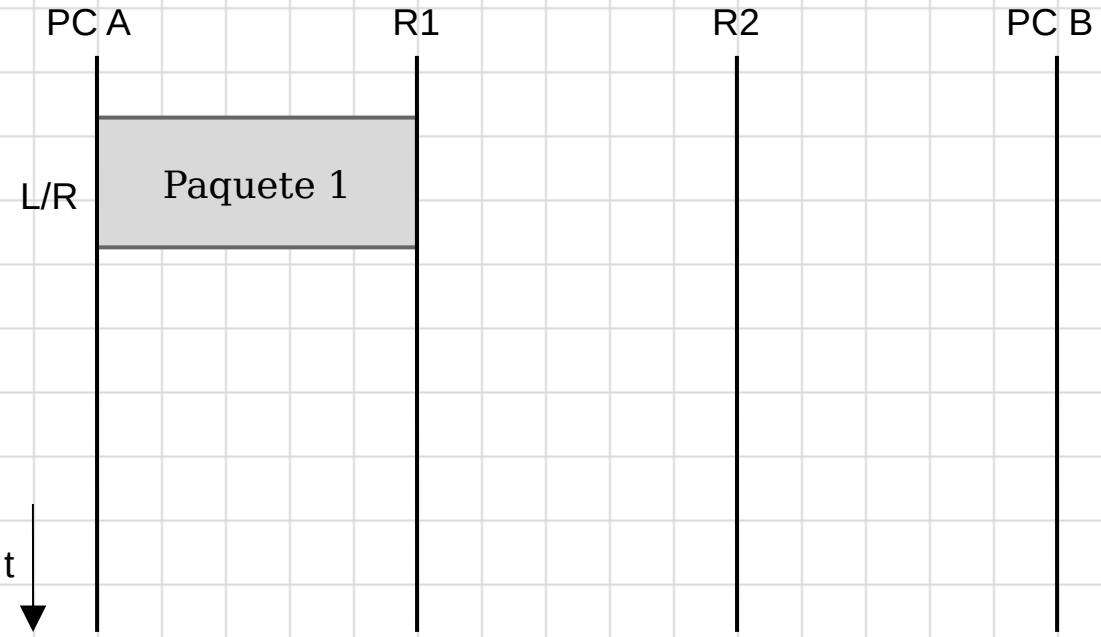


Se considera que son conexiones persistentes.

$RTT_{DNS} = 5$ ms
$RTT_1 = 10$ ms
$RTT_2 = 6$ ms
$T_{descarga} = RTT_{DNS} + 2 \times RTT_1 + RTT_{DNS} + 2 \times RTT_2 = 42$ ms

Curso 2022/23

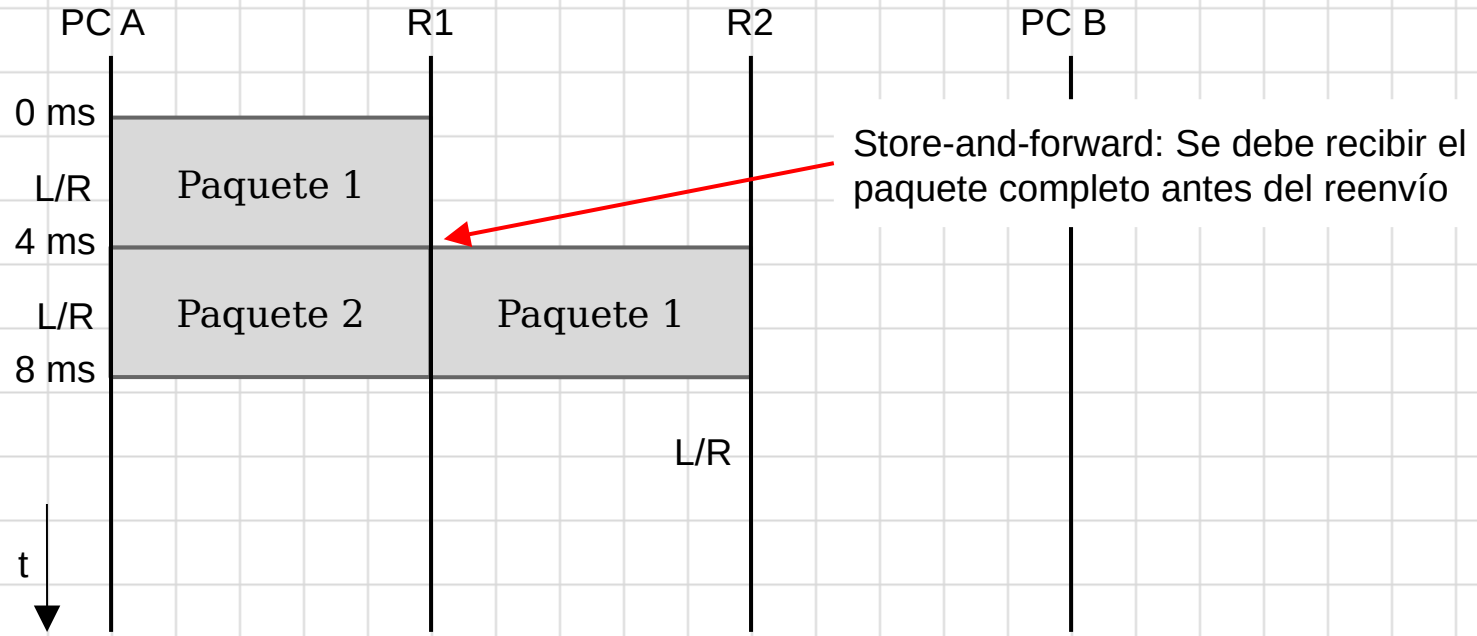
Julio 2023 – P1, ap. a)



Tiempo que se tarda en transmitir un paquete de L bits sobre un enlace con ancho de banda R (bps):

$$d_{trans} = \frac{L}{R} = \frac{4000\ bits}{1\ Mbps} = \frac{4 \times 10^3 bits}{1 \times 10^6 bps} = 4\ ms$$

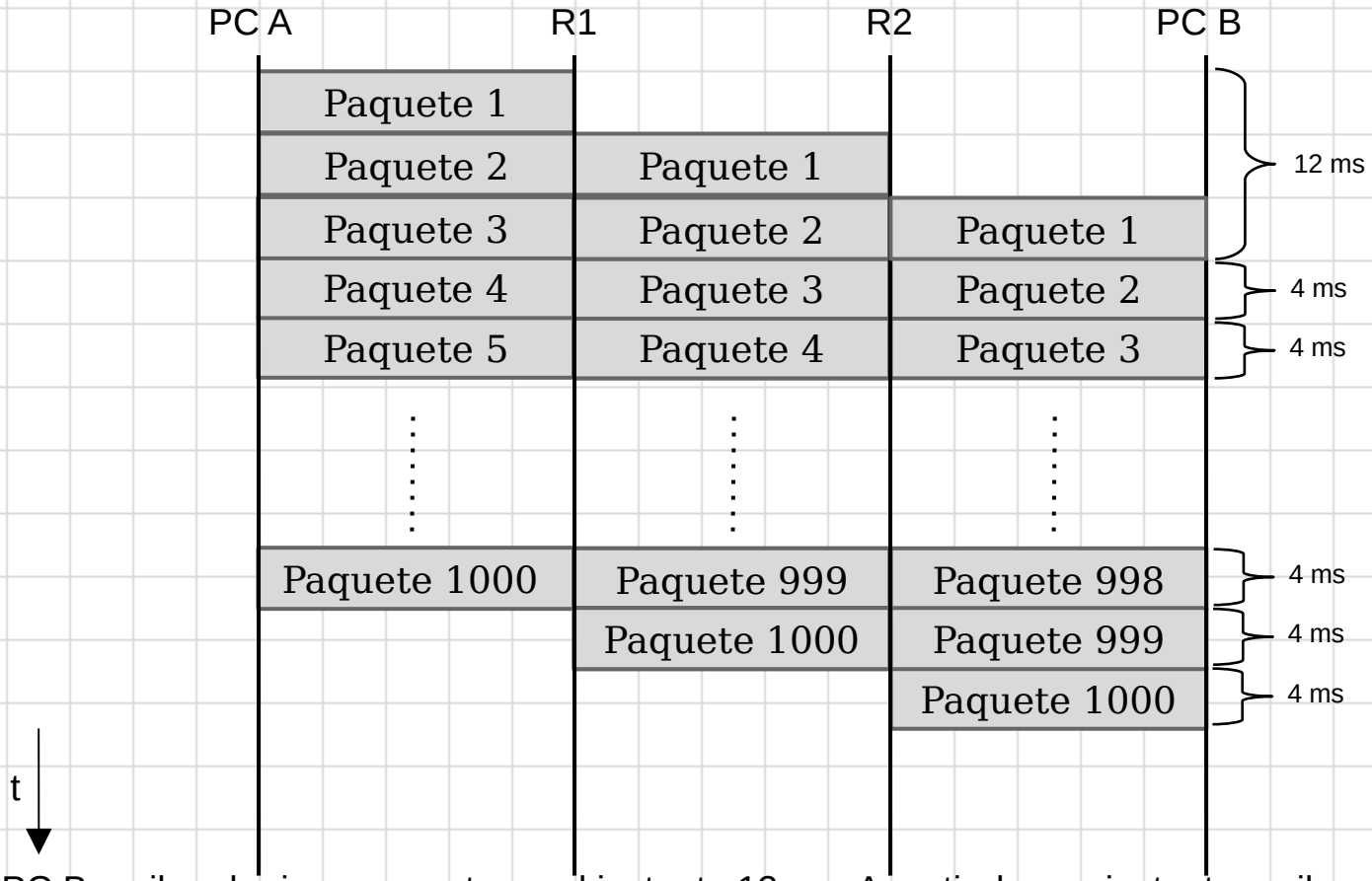
Julio 2023 – P1, ap. b)



El router R1 habrá recibido el segundo paquete completo en el instate 8 ms.

(Nota: tal y como se comenta en el enunciado del problema hay que ignorar el resto de retardos)

Julio 2023 – P1, ap. c)



El PC B recibe el primer paquete en el instante 12 ms. A partir de ese instante recibe un paquete cada 4 ms.

$$t_{completo} = 12\ ms + 999 \times 4\ ms = 4008\ ms$$