



CÁTEDRA:

REDES Y TRANSMISIÓN DE DATOS

ENUNCIADO DE TRABAJOS DE LABORATORIO

Año 2018

Versión 1.12

Docentes

Profesor Adjunto: **Mg. Ing. Ricardo Antonio López**
Jefe de Trabajos Prácticos: **Lic. Nahuel Defossé**

INDICE

TRABAJOS DE LABORATORIO.....	3
Objetivo general de los Trabajos Prácticos de Laboratorio.....	3
Características.....	3
Forma de aprobación.....	3
Lista de Trabajos de Laboratorio.....	4
Trabajo de Laboratorio 1.....	5
Capa de Aplicación.....	5
Parte 1: HTTP. Telnet. SMTP.....	5
HTTP.....	5
Interacciones HTTP Básicas.....	5
Interacciones HTTP condicionales.....	6
Descarga de documentos grandes.....	7
Documentos HTML con Objetos Empotrados.....	8
Autenticación HTTP.....	8
TELNET AL PUERTO 80.....	10
Utilidades de línea de comandos.....	10
Hay ciertas herramientas de línea de comandos en GNU/Linux que pueden resultar útiles para.....	10
experimentar y/o depurar aplicaciones cliente-servidor, especialmente las desarrolladas.....	10
directamente sobre el interfaz de sockets.....	10
Ejemplos de servicios con estado y sin estado.....	12
Parte 2: Programación Cliente / Servidor utilizando la interfaz sockets.....	14
Código base.....	14
Tareas a Realizar.....	15
Parte 3: NMAP.....	15
Introducción.....	15
1. Enumerar equipos de la red y sus servicios.....	15
Trabajo de Laboratorio 2.....	17
Capa de Transporte.....	17
Parte 1: TCP Fundamentos.....	17
Parte 2: TCP Análisis de Segmentos.....	19
Parte 3: TCP Análisis de Tramas.....	25
Trabajo de Laboratorio 3.....	27
Capa de Red.....	27
Parte 1: IP Fundamentos -Subnetting.....	27
Parte 2: IP Fragmentación – ICMP - Encaminamiento.....	29
Parte 3: Ip Routing - NAT.....	32
Trabajo de Laboratorio 4.....	37
Capa de Enlace y Física.....	37
Parte 1: MAC – ARP - Ethernet.....	37

TRABAJO DE LABORATORIO

Objetivo general de los Trabajos Prácticos de Laboratorio.

- Conocimiento y realización de trabajos sobre protocolos de Aplicación.
- Conocimiento de la capa de transporte y su problemática específica.
- Manejo de la capa de RED. Subnetting, ruteo, NAT. IPV6.
- Conocimiento de la capa de enlace y Física.

Características

El Informe deberá reunir las siguientes características:

1. Cada Grupo presentará su informe a efectos de su calificado por el profesor. Los trabajos que no reúnan los requisitos mínimos serán devueltos para su corrección.
2. Deberá ser presentado a la cátedra confeccionado en grupos **no mayores de dos alumnos**, con **discusión individual** por alumno.
3. Para su preparación e impresión, el trabajo práctico deberá ser entregado de la siguiente forma:
 - que deberá identificar a los integrantes del grupo y contener la firma de los mismos.
 - Se deberá trabajar sobre el enunciado de cada punto de la práctica, **conservando su enunciado previo a la solución**.
 - Toda la bibliografía utilizada deberá ser referenciada indicando título y autor, en una sección dedicada a tal efecto.
 - De existir algún código de programación, de deberá adjuntar el programa de aplicación que implementa la solución. El código fuente debe estar debidamente comentado. La solución debe ser desarrollada utilizando el lenguaje de programación indicado. Se debe incluir la demostración de prueba de la solución implementada.
 - El formato de entrega será **Notebooks de Jupyter/IPython** utilizando **texto enriquecido** para los enunciados y las respuestas. El código fuente deberá estar correctamente adaptado para la prueba por parte del docente o correctamente formateado con la coloración adecuada. Para utilizar este software, usar Anaconda, Microsoft Azure Notebooks o la instalación expuesta en el aula.
 - Todos los prácticos deberán estar munidos de su archivo de captura pcap/pcap-ng.

Forma de aprobación

Se tendrá en cuenta para la aprobación del trabajo práctico y los integrantes del grupo:

- Estructura general de la presentación, su legibilidad y facilidad de lectura y comprensión.
- Contenido del informe y el uso de la información técnica para elaborarlo.
- Evaluación del grupo como un todo y a cada uno de sus integrantes.
- Funcionamiento de la aplicación desarrollada. Se evaluará si la funcionalidad cumple con lo solicitado.

En caso de no verificarse algunas de las condiciones enunciadas, el trabajo práctico se considerará desaprobado.

Lista de Trabajos de Laboratorio

PRÁCTICA N° 1 – Capa de Aplicación.

PRÁCTICA N° 2 – Capa de Transporte.

PRÁCTICA N° 3 – Capa de Red.

PRÁCTICA N° 4 – Capas de Enlace y Física.

Trabajo de Laboratorio 1

Capa de Aplicación

Parte 1: HTTP. Telnet. SMTP

HTTP.

Interacciones HTTP Básicas.

Para comenzar a analizar el funcionamiento de **HTTP** vamos a comprobar cómo descarga el navegador una página web sencilla que tan sólo constará de un documento **HTML** de pequeño tamaño sin objetos incrustados (imágenes, vídeos, etc.). Para dicha tarea emplearemos el analizador de red **Wireshark**.

Vamos a seguir los siguiente pasos:

- Arranque un navegador Web.
- Arranque el analizador de red **Wireshark**. Para que el analizador de red sólo nos muestre los mensajes del protocolo **HTTP** introduciremos la cadena '**http**' (sin las comillas) en la ventana de especificación de filtros de visualización (display-filter). Si no hiciéramos esto veríamos todo el tráfico de red que es capaz de capturar nuestra tarjeta de red.
- Espere aproximadamente un minuto (ya veremos por qué), y comience a capturar paquetes empleando **Wireshark** (Menú *Capture* -> *Interfaces* -> *Start*).



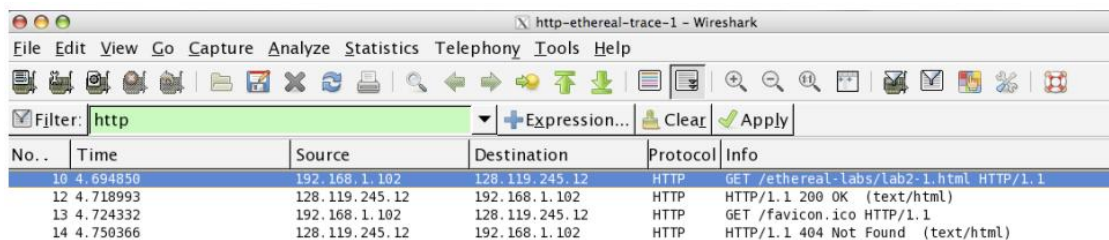
- Introduzca la siguiente dirección en la barra de direcciones de su navegador:

http://IPMaquinaProfesor/HTTP-wireshark-file1.html

Su navegador mostrará una página que sólo constará de una línea de texto.

- Detenga la captura de paquetes en **Wireshark** (Menú *Capture* -> *Interfaces* -> *Stop*).

La ventana de **Wireshark** debería mostrar un aspecto similar a la que se muestra en la siguiente figura:



El ejemplo de la figura anterior muestra una ventana con un listado de paquetes que contiene

los dos mensajes **HTTP** capturados: el mensaje **GET** (desde su navegador hacia el servidor Web) y el mensaje de respuesta desde el servidor hacia su navegador.

Justo debajo de la ventana que muestra el listado de mensajes, encontrará una ventana que muestra el contenido del paquete que esté seleccionado (el que aparece resaltado en el listado anterior).

Dado que los mensajes **HTTP** se encapsulan en segmentos **TCP** que a su vez van dentro de datagramas **IP** que son transportados en una trama (en este caso **Ethernet**), **Wireshark** muestra también la información de la trama, del datagrama y del segmento.

Como en este caso tan sólo estamos interesados en **HTTP** ocultaremos toda esa información. Para ello compruebe que al lado de la información de trama, **Ethernet**, **IP** y **TCP** aparece un signo '+' (que significa que hay información oculta que no es mostrada). Compruebe también que precediendo la línea de **HTTP** aparece un signo '-'.

Analice la información de los mensajes **HTTP** de **petición** y **respuesta**, y conteste las preguntas que se plantean. En la respuesta a las preguntas debe indicar las porciones de los mensajes de **petición** y **respuesta** en las que basa su respuesta:

1a. ¿Qué versión de **HTTP** emplea su navegador?

1b. ¿Qué versión de **HTTP** ejecuta el servidor?

2a. ¿Qué idiomas indica su navegador al servidor que está dispuesto aceptar en la respuesta?

2b. ¿Qué relación tiene esta información con el lenguaje que emplea en su navegador y sistema operativo?

3a. ¿Cuál es la dirección **IP** de su ordenador?

3b. ¿Y del servidor Web al que está accediendo?

4a. ¿Cuál es el código de estado devuelto a su navegador por el servidor?

4b. ¿Cuál es el significado de ese código de estado?

5. ¿Cuándo fue modificado por última vez el fichero **HTML** que el servidor te está devolviendo?

Interacciones HTTP condicionales.

La mayoría de navegadores Web guardan una copia en **caché** de parte de los archivos a los que acceden, lo que posibilita la realización de peticiones **GET condicionales**. En este apartado vamos a comprobar el funcionamiento de este mecanismo. Para poder apreciarlo adecuadamente es necesario que nos aseguremos que la **caché** de nuestro navegador esté vacía. A continuación, haga lo siguiente:

- a) Arranque el navegador y cerciórese de que la caché de su navegador esté vacía.
- b) Arranque el analizador de red **Wireshark** y comience a capturar paquetes.
- c) Introduzca la siguiente **URL** en la barra de direcciones de su navegador:

http:// 174.138.32.232 /HTTP-wireshark-file2.html

En la ventana del navegador verá un fichero **HTML** sencillo de tan varias líneas.

Acto seguido, **actualice la página en el navegador**.

Detenga la captura de paquetes en Wireshark e introduzca '**http**' en la ventana de especificación de filtros de visualización de modo que tan sólo se muestren los mensajes **HTTP** en la ventana de listado de paquetes.

Responda a las siguientes preguntas:

8. Inspeccione los contenidos de la primera petición **GET** de su navegador hacia el servidor. ¿Observa alguna línea '**If-Modified-Since**' en la petición **GET**? Indique el contenido de dicha línea.
9. Analice el contenido de la respuesta del servidor. ¿Contiene dicha respuesta el fichero **HTML** que se pedía?
10. Ahora analice el contenido de la segunda petición **GET** desde su navegador al servidor. ¿Ve alguna línea '**If-Modified-Since**' en dicha petición? Si es así, ¿qué información aparece a continuación de dicha cabecera?
11. ¿Cuál es el código de estado **HTTP** y el mensaje devuelto por el servidor en respuesta a este segundo **GET**? ¿Devuelve el servidor explícitamente el contenido del fichero? Explique qué ha sucedido.

Descarga de documentos grandes.

En todos los ejemplos hasta ahora, los documentos que hemos descargado eran documentos **HTML** sencillos.

Comprobemos qué sucede si descargamos un fichero **HTML** grande (sin objetos dentro del **HTML**). Haga lo siguiente:

4. Arranque su navegador web.
5. Arranque el analizador de paquetes Wireshark y comience a capturar paquetes.
6. Introduzca la siguiente URL en el navegador:

<http://174.138.32.232/HTTP-wireshark-file3.html>

El navegador debería mostrar un documento **HTML** de cierta longitud.

- Detenga la captura de paquetes en **Wireshark** e introduzca '**http**' en la ventana de especificación de filtros de visualización de manera que tan sólo se muestren los mensajes **HTTP**.

En la ventana de listado de paquetes debería ver un mensaje **HTTP** del tipo **GET** seguido de una respuesta multi-paquete a este mensaje. Esta respuesta multipaquete merece una explicación.

Han visto en la teoría que el mensaje de respuesta **HTTP** consta de una línea de estado seguido de líneas conteniendo distintas cabeceras, seguidas de una línea en blanco, seguida del cuerpo de la entidad. En el caso de **GET**, este cuerpo es el fichero **HTML** pedido.

Para nuestro caso el fichero *HTTP-wireshark-file3.html*, es bastante grande, los ~20Kbytes de los que consta no caben en un único mensaje **TCP**. Esto obliga a que la respuesta **HTTP** se divida en varias partes, viajando cada una de ellas en un segmento **TCP** separado (recomendamos consultar la bibliografía para ver este tema). **Wireshark** registra cada segmento **TCP** como un paquete independiente. El hecho de que una única respuesta **HTTP** se divida en múltiples paquetes **TCP** se indica por parte de **Wireshark** mediante el mensaje "*Continuation*" en cada uno de los segmentos **TCP** que siguen al primero. Esto es únicamente un mecanismo que emplea Wireshark para indicar esto, no existe un mensaje "*Continuation*" ni en **HTTP** ni en **TCP**.

Responda a las siguientes preguntas:

12. ¿Cuántos mensajes de petición **GET** han sido enviados por el navegador?
13. ¿Cuántos segmentos **TCP** con datos han sido necesarios para una única respuesta **HTTP**?
14. ¿Cuál es el código de estado y el mensaje asociado con la respuesta a la petición **GET**?

Documentos HTML con Objetos Empotrados

Una vez que hemos visto la forma en que Wireshark muestra el tráfico de paquetes capturados para mensajes HTML grandes, vamos a analizar qué sucede cuando nuestro navegador descarga un fichero con objetos empotrados, por ejemplo un fichero que incluye otros objetos (en el ejemplo que veremos, imágenes) que se almacenan en otro u otros servidores.

Lleve a cabo las siguientes acciones:

1. Inicie el navegador web y asegúrese que ha vaciado la caché tal y como hemos explicado anteriormente.
2. Arranque Wireshark y comience a capturar paquetes.
3. Introduzca la siguiente URL en el navegador:

<http://174.138.32.232/HTTP-wireshark-file4.html>

El navegador debería mostrar un fichero HTML con dos imágenes. Estas dos imágenes se referencia en el fichero HTML, es decir, las dos imágenes no están contenidas en el HTML sino que lo que aparece en el HTML son las URLs donde se ubican las imágenes. Por ejemplo, la primera imagen aparece referenciada del siguiente modo empleando la etiqueta html **img**.

``

Si copia la URL que aparece a continuación de "**src**" y la pega en la barra de direcciones del navegador, comprobará como el navegador muestra una página que tan sólo contiene la primera imagen de las dos que aparecen en la página. Tal y como hemos visto, el navegador tendrá que descargar esas imágenes de los sitios web indicados en el HTML.

Detenga la captura de paquetes de Wireshark e introduzca 'http' en la ventana de especificación de filtros de visualización de modo que sólo se muestren los mensajes HTTP.

Responda a las siguientes preguntas:

15. ¿Cuántos mensajes **GET** ha enviado el navegador? ¿A qué direcciones se han enviado esos mensajes?
16. ¿Puede determinar si las imágenes han sido descargadas sucesivamente (se pide una imagen, se descarga y después se pide la otra) o en (se lanzan las dos descargas de forma simultánea)?

Autenticación HTTP.

Finalmente vamos a tratar de visitar un sitio web protegido mediante una clave y vamos a examinar la secuencia de mensajes http intercambiados para dicho sitio.

La URL: *http://174.138.32.232/protected_pages/HTTP-wireshark-file5.html*

está protegida mediante una clave. El nombre de usuario es *'wireshark'* (sin las comillas) y la clave es *'network'* (sin las comillas una vez más). Vamos a acceder a este sitio 'protegido' mediante esta pareja usuario/contraseña.

Haga lo siguiente:

- Asegúrese que la caché del navegador está vacía, tal y como hemos explicado antes, y cierre el navegador. A continuación arranque el navegador de nuevo.
- Arranque Wireshark y comience a capturar paquetes
- Introduzca la siguiente URL en el navegador:

http://174.138.32.232/protected_pages/HTTP-wireshark-file5.html

Introduzca el nombre de usuario y la contraseña.

- Detenga la captura de paquetes de Wireshark e introduzca **'http'** en la ventana de especificación de filtros de visualización de modo que sólo se muestren los mensajes HTTP.

Examine la salida de Wireshark. **Se recomienda leer primero el siguiente documento sobre autenticación HTTP:** [http://frontier.userland.com/stories/storyReader\\$2159](http://frontier.userland.com/stories/storyReader$2159)

Responda a las siguientes preguntas:

17. ¿Cuál es el código de respuesta y el mensaje del servidor en respuesta al mensaje GET inicial de su navegador?
18. Cuando su navegador envía el mensaje HTTP GET por segunda vez, ¿qué nuevo campo se incluye en el mensaje GET?
19. Explique la codificación Base64.

El nombre de usuario (wireshark) y la contraseña (network) introducidas se codifican en la cadena de caracteres (d2lyZXNoYXJrLXN0dWRlbnRzOm5ldHdvcm90= o algo así) que aparece a continuación de la cabecera **"Authorization: Basic"** en el mensaje **GET** del cliente. Aunque pudiera parecer que se envían cifrados, simplemente están codificados en un formato conocido como Base64. Puede comprobar esto accediendo a cualquier codificador de **Base64** web, donde se vería que se concatenan usuario y contraseña codificados en **Base64**.

Cualquier persona con acceso a una utilidad como Wireshark podría capturar paquetes de la red y trasladarlos de **Base64** a **ASCII**. Esto debería dejar claro que el empleo de sistemas sencillos de autenticación en HTTP como los descritos aquí no son seguros a menos que se tomen medidas adicionales.

TELNET AL PUERTO 80.

Telnet es una aplicación que también utiliza TCP para el transporte de sus mensajes.

El servidor telnet ejecuta los comandos remotos enviados por el cliente y devuelve la respuesta resultante de la ejecución de los mismos. Si solicitamos una conexión TCP al puerto 80 de un servidor web mediante telnet, engañaremos al servidor web, que pensará que somos un navegador y, una vez establecida la conexión, podremos escribirle un mensaje HTTP directamente al servidor web.

- Desde un terminal, ejecuta el siguiente comando:

```
%> telnet <IP DEL PROFESOR> 80
```

- Después escribe el siguiente mensaje de petición HTTP:

```
GET /index.html HTTP/1.1  
Host: localhost
```

Pulsar dos veces INTRO al terminar la línea para enviar el mensaje. Deberías obtener la página solicitada como respuesta.

Intenta ahora volver a conectarte y envíale un método erróneo, por ejemplo:

```
JET /index.html HTTP/1.1  
Host: localhost
```

20. ¿Qué mensaje recibes como respuesta?

Intenta ahora solicitar un objeto que no existe, por ejemplo:

```
GET /ittex.html HTTP/1.1  
Host: 174.138.32.232
```

21. ¿Qué mensaje recibes como respuesta ahora?

Utilidades de línea de comandos.

Hay ciertas herramientas de línea de comandos en GNU/Linux que pueden resultar útiles para experimentar y/o depurar aplicaciones cliente-servidor, especialmente las desarrolladas directamente sobre el interfaz de sockets.

Comando nc/netcat.

netcat (**nc**) es un comando que permite acceder a puertos TCP o UDP de la máquina local o de otras remotas.

También permite quedar a la escucha en un puerto dado (TCP o UDP) de la máquina local.

Funcionamiento

- Cuando funciona como cliente, **nc** crea un socket para conectarse al puerto indicado de la máquina destino.

`$ nc maquina_destino puerto_destino`

- La conexión permanecerá abierta mientras no la finalice el servidor o el cliente **nc** (control+C).
- Cuando funciona como servidor (modo escucha [opción **-l**, listen]), abre un socket en la máquina local que queda a la escucha en el puerto indicado:

`$ nc -l -p puerto_escucha`

Esta operación solo escucha una conexión.

```
→ ~ nc -l -p 9090
HOLA
→ ~
[0] 0:~* "usuario-virtual-machi" 12:37 02-abr-18
```

```
→ ~ echo "HOLA" | nc localhost 9090
→ ~
```

- ncat** (instalado con `sudo apt-get install nmap`) o **socat** permiten conexiones sucesivas e incluso concurrentes en modo servidor (modo también conocido como escucha o pasivo).¹

```
→ ~ ncat -l -k -p 9090
terminal 1
terminal 2
[0] 0:nc* "usuario-virtual-machi" 12:45 02-abr-18
```

```
→ ~ echo "HOLA" | nc localhost 9090
→ ~ nc localhost 9090
terminal 1
[0] 0:nc* "usuario-virtual-machi" 12:45 02-abr-18
```

```
→ ~ nc localhost 9090
[0] 0:nc* "usuario-virtual-machi" 12:45 02-abr-18
```

```
→ ~ socat - TCP-LISTEN:9090,fork
hola
hola
1
2
[0] 0:nc* "usuario-virtual-machi" 12:50 02-abr-18
```

```
→ ~ nc localhost 9090
hola
1
[0] 0:nc* "usuario-virtual-machi" 12:50 02-abr-18
```

```
→ ~ nc localhost 9090
hola
2
[0] 0:nc* "usuario-virtual-machi" 12:50 02-abr-18
```

¹ Las imágenes a continuación fueron generadas con **tmux**

En ambos casos, una vez establecida la conexión, **nc** envía a través del socket creado todo lo que reciba por la entrada estándar y envía a la salida estándar lo que le llegue por el socket.

Opciones interesantes:

-u usa el modo UDP (por defecto son conexiones TCP)

-c comando / **-e ejecutable** ejecuta un comando/programa una vez iniciada la conexión cuyas entrada y salida estándar están redirigidas a la conexión establecida.

Pruebas a realizar

Abrir una sesión **nc** en modo escucha desde un terminal en un puerto no privilegiado. En otro terminal abrir otra sesión **nc** que se conecte al primero. Comprobar que el funcionamiento es el descrito.

Repetir el ejercicio entre dos máquinas.

Comando netstat.

netstat es un comando que ofrece información (uso, estado, etc) sobre las conexiones de la máquina propia, tanto entrantes como salientes, además de otras informaciones sobre la red.

Informa tanto de conexiones del dominio de Internet (locales y remota) como de conexiones del dominio UNIX (entre procesos locales).

La información típica que muestra para cada conexión de red es:

- tipo de protocolo
- dirección+puerto local
- dirección+puerto remoto
- estado de la conexión (para TCP): establecida, cerrada, esperando cierre, en espera, etc

Opciones interesantes:

- **-a** muestra información sobre todas las conexiones/sockets (tanto TCP y UDP)
- **-p** muestra los procesos que están usando las conexiones.
- **-l** muestra información sobre las conexiones/sockets en modo escucha
- **-e** información en formato extendido
- **-t** sólo conexiones TCP, **-u** sólo conexiones UDP.

Pruebas a realizar.

Comprobar los distintos parámetros de **netstat** e identificar las conexiones abiertas en la máquina local.

Comprobar las conexiones abiertas en los ejemplos de comunicación entre sesiones **nc** de la sección anterior, identificar los puertos cliente de la sesión **nc**.

Ejemplos de servicios con estado y sin estado

Servicio sin estado. HTTP.

Es el ejemplo típico de servicio sin estado, implementa un esquema una petición → una respuesta.

Acceso a un servidor WEB.

Comprobar el diálogo HTTP desde el punto de vista de un cliente.

1. Ejecutar **nc** en modo conexión con el puerto 80 de un servidor web accesible.

```
$ nc 174.138.32.232 80
```

2. Escribir la siguiente petición.

```
GET /index.html HTTP/1.1  
Host: localhost
```

Simulación de un servidor WEB en el puerto 8080

Comprobar el diálogo HTTP desde el punto de vista del servidor.

1. Ejecutar **nc** en modo escucha en el puerto 8080.

```
$ ncat -k -l -p 8080
```

2. Desde un navegador web escribir `http://localhost:8080/index.html`

- **nc** muestra la petición HTTP realizada por el navegador

3. Responder a la petición desde **ncat**.

```
HTTP/1.1 200 OK
```

```
Content-Type: text/html  
Content-Length: 51
```

```
<html><body> hola  </body> </html>
```

- Comprobar lo que sucede en la sesión **nc** que simula el servidor web.

4. Realizar otra petición desde el navegador (recargar página) y enviarle una nueva respuesta desde **ncat**.

```
HTTP/1.1 200 OK  
Content-Type: text/html  
Content-Length: 71
```

```
<html><body> hola <a href="principal.html"> enlace </a> </body> </html>
```

- Pinchar en el enlace y comprobar que sucede.

Servicio con estado. POP3 y SMTP

Los protocolos POP3 (servicio de correo entrante – protocolo de oficina de correo) y SMTP (servicio de correo saliente – protocolo simple de transferencia de correo) son ejemplos de servicios con estado. En función de los mensajes previos enviados por el cliente, como parte de una conexión establecida previamente, las respuestas del servidor varían.

Típicamente, el comportamiento de este tipo de servicios con estado estará gobernado por una máquina de estados (o un mecanismo equivalente) exclusiva para cada cliente. Esa información permite que el servidor pueda llevar traza de las peticiones recibidas y del estado en el que se encuentra el diálogo con el cliente.

Para la ejercitación para el año en curso, se ha dispuesto la siguiente máquina virtual en la nube: 174.138.32.232. Para evitar el abuso por parte de *bots* en búsqueda de *open relays* se han substituido los puertos standard 25 por 2500 y 110 por 1100.

Conectar con un servidor SMTP (puerto 25)

Elegir un servidor SMTP donde el alumno tenga acceso y simular el diálogo de un cliente SMTP a la hora de enviar un e-mail.

1. Ejecutar **nc** en modo conexión con el puerto 25 del servidor SMTP²

\$ telnet 174.138.32.232 2500

2. Obtener información sobre el servidor SMTP.

Escribir el siguiente mensaje en la sesión nc:

HELO <nombre o IP del cliente>

3. Enviarse un e-mail a si mismo.

Escribir los siguientes mensajes en la sesión nc:

MAIL FROM: login@aaa.aaa.com

RCPT TO:login@localhost

DATA

Subject: asunto del mensaje

mensaje a enviar

mensaje a enviar

- a) "MAIL FROM:" indica quien envía el mensaje
- b) "RCPT TO:" indica quien es el destinatario (si fueran varios se repite para cada uno de ellos)
- c) "DATA" indica el comienzo del contenido del mensaje.
- d) El fin del mensaje se marca con "." (+ retorno de carro)

4. Cerrar la conexión con el servidor SMTP

Escribir el siguiente mensaje en la sesión **nc**:

QUIT

5. Comprobar la recepción del mensaje.

² El software que se ejecuta se encuentra disponible en <https://github.com/UNPSJB/unp-rtdtp1/> y puede ejecutarse si se cuenta con el servicio Docker y la utilidad docker-machine

Conectar con un servidor POP3 (puerto 110)

Utilizar el servidor POP3 expuesto en el puerto 1100 y realizar el diálogo de un cliente POP3 a la hora de descargar los mensajes del buzón del usuario (es necesario haber completado el punto anterior para poder listar mensajes).

1. Ejecutar nc en modo conexión con el puerto 110 del servidor POP3

```
$ telnet 174.138.32.232 1100
```

2. Autenticarse frente al servidor.

Escribir los siguientes 2 mensajes en la sesión nc (envío del nombre de usuario + envío de la clave)

```
USER miusuario
PASS mipassword
```

Aviso: el tráfico irá en claro y el login y el password serán visibles para cualquiera que escuche en ese enlace.

3. Gestionar los mensajes del buzón

Escribir la siguiente secuencia de mensajes en la sesión nc (ver estado del buzón, obtener la lista de mensajes pendientes de descargar, descargar el primer mensaje, cerrar la conexión POP3)

```
STAT
LIST
RETR 1
QUIT
```

- Comprobar las respuestas y resultados mostrados en el cliente nc

Parte 2: Programación Cliente / Servidor utilizando la interfaz sockets.
Código base (disponible también en <https://gist.github.com/D3f0/a6259043846896205675c10c1aec56e4>)

```
/* PrimerServidorTCP.c
Servicio: Las cadenas de texto recibidas de un Cliente son enviadas a la salida estándar.
Nota: Por simplicidad del código no se realiza ningún tipo de control de errores. No obstante el servidor es totalmente funcional.
*/
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>

#define PORTNUMBER 12345
```

```
int main(void){
    char buf[10];
    int s, n, ns, len;
    struct sockaddr_in direcc;

    s = socket(AF_INET, SOCK_STREAM, 0);

    bzero((char *) &direcc, sizeof(direcc));
    direcc.sin_family = AF_INET;
    direcc.sin_port = htons(PORTNUMBER);
    direcc.sin_addr.s_addr = htonl(INADDR_ANY);

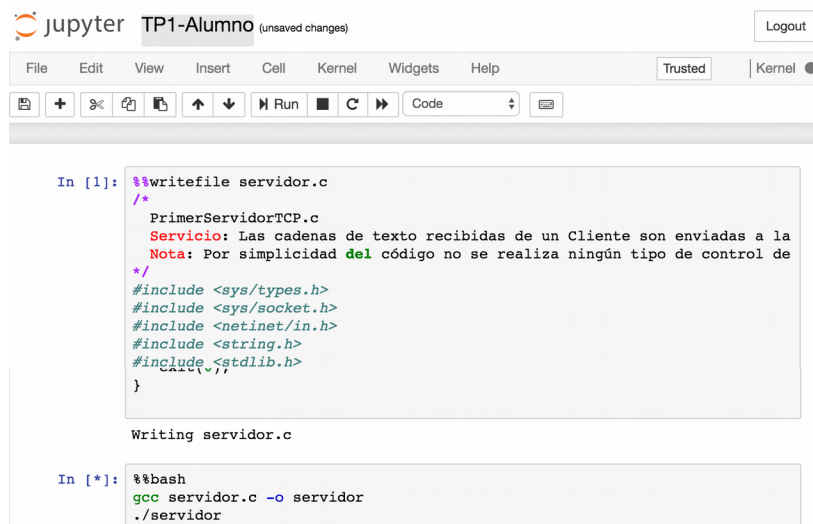
    len = sizeof(struct sockaddr_in);
    bind(s, (struct sockaddr *) &direcc, len);
    listen(s, 5);
    ns = accept(s, (struct sockaddr *) &direcc, &len);

    while ((n = recv(ns, buf, sizeof(buf), 0)) > 0)
        write(1, buf, n);

    close(ns); close(s);
    exit(0);
}
```

Tareas a Realizar.

Grabe el contenido del archivo en una celda que comience con `%%writefile servidor.c` y luego en otra celda que comience con `%%bash`, compile el servidor (`gcc -o servidor servidor.c`) y luego ejecútelo (`./server`). Abra otra terminal y contactese con el servidor usando el cliente *telnet* (`#> telnet localhost 12345`). Verifique el funcionamiento del mismo y luego cierre la sesión de *telnet*.



```
In [1]: %%writefile servidor.c
/*
PrimerServidorTCP.c
Servicio: Las cadenas de texto recibidas de un Cliente son enviadas a la
Nota: Por simplicidad del código no se realiza ningún tipo de control de
*/
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <stdlib.h>
}

Writing servidor.c

In [*]: %%bash
gcc servidor.c -o servidor
./servidor
```


1. ¿Que sucede con el Servidor? ¿Por qué?
2. Modifique el servidor para que no finalice.
3. Modifique el servidor para que el servicio que presta se implemente en una función separada del cuerpo principal del programa (*main()*).
4. Modifique el servidor para que preste el servicio de **echo** (vea RFC862).
5. Modifique el servidor para que preste el servicio de **character generator** (vea RFC864).
6. Desarrolle un cliente que se comuniquen con el servidor.
7. Modificar el Servidor del punto anterior para que cambie el servicio por uno que devuelva la suma de dos parámetros enteros o una leyenda de error en parámetros si estos no son enteros.

Tips: Usar la función `sscanf()` y `snprintf()`. Crear el cliente correspondiente.

8. Transformar el Servidor iterativo TCP de Eco del punto 4), en un servidor iterativo de Eco con UDP. Crear el cliente correspondiente.

Parte 3: NMAP

Introducción.

El práctico consistirá en el uso de la herramienta de escaneo de puertos **nmap** para obtener información de los equipos y servicios de la red (puede utilizar Kali o instalarlo a través de `apt-get install nmap`).

nmap implementa diversas técnicas para extraer información de los equipos que forman parte de una red y para identificar los puertos y servicios que están disponibles en distintas máquinas. Algunos de los métodos disponibles realizan el escaneo sin dejar rastro, mientras que otros dejarán un rastro en los ficheros de log de las máquinas analizadas.

Página de nmap: <http://www.nmap.org>

Más información: <http://es.wikipedia.org/wiki/Nmap>

Manual en español: <http://nmap.org/man/es/>

Tutorial en inglés: <http://www.nmap-tutorial.com>

1. Enumerar equipos de la red y sus servicios.

A las máquinas de cada grupo las haremos funcionar como *observador*, para ello:

Debe determinar la familia de direcciones IP en la cual su interface de red está conectada. Este tipo suele ser 192.168.0.x, pero puede variar. Se ha observado en el Laboratorio 2 que las IP son de la familia 192.168.88.x. Para esto debe utilizar la utilidad `ifconfig` o `ip addr`, a continuación se muestra la salida del comando `ifconfig` sobre Ubuntu 16.04.3:

```
user@usuario-virtual-machine: /mnt/hgfs/Universidad/RTD/2018/Ejemplo/linux
jupyter notebook user@usuario-virtual-machine: /mnt/hgfs...
→ linux ifconfig
docker0  Link encap:Ethernet direcciónHW 02:42:7f:1a:17:c9
         Direc. inet:172.17.0.1 Difus.:172.17.255.255 Másc:255.255.0.0
         Dirección inet6: fe80::42:7fff:fe1a:17c9/64 Alcance:Enlace
         ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST MTU:1500 Métrica:1
         Paquetes RX:0 errores:0 perdidos:0 overruns:0 frame:0
         Paquetes TX:7606 errores:0 perdidos:0 overruns:0 carrier:0
         colisiones:0 long.colaTX:0
         Bytes RX:0 (0.0 B) TX bytes:1255827 (1.2 MB)

ens33    Link encap:Ethernet direcciónHW 00:0c:29:e6:41:12
         ACTIVO DIFUSIÓN MULTICAST MTU:1500 Métrica:1
         Paquetes RX:0 errores:0 perdidos:0 overruns:0 frame:0
         Paquetes TX:0 errores:0 perdidos:0 overruns:0 carrier:0
         colisiones:0 long.colaTX:1000
         Bytes RX:0 (0.0 B) TX bytes:0 (0.0 B)

ens38    Link encap:Ethernet direcciónHW 00:0c:29:e6:41:1c
         Direc. inet:192.168.1.215 Difus.:192.168.1.255 Másc:255.255.255.0
         Dirección inet6: fe80::3739:578d:50d9:f64b/64 Alcance:Enlace
         ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST MTU:1500 Métrica:1
         Paquetes RX:274821 errores:0 perdidos:2 overruns:0 frame:0
         Paquetes TX:62712 errores:0 perdidos:0 overruns:0 carrier:0
         colisiones:0 long.colaTX:1000
```

a) Lanzar un escaneado *Ping Sweeping* [opción -sP] para identificar, mediante Ping, las máquinas que componen la red

```
nmap -sP 192.168.0.0/24
```

```
In [2]: %%bash
        nmap -sP 192.168.1.0/24

Starting Nmap 7.60 ( https://nmap.org ) at 2018-03-27 12:36 -03
Nmap scan report for Wifi (192.168.1.1)
Host is up (0.0019s latency).
Nmap scan report for morrongo.fibertel.com.ar (192.168.1.55)
Host is up (0.00070s latency).
Nmap scan report for 192.168.1.84
Host is up (0.017s latency).
Nmap scan report for 192.168.1.88
Host is up (0.077s latency).
Nmap scan report for HP98255 (192.168.1.143)
Host is up (0.016s latency).
Nmap scan report for 192.168.1.168
Host is up (0.11s latency).
Nmap scan report for usuario-virtual-machine.fibertel.com.ar (192.168.1.215)
Host is up (0.0038s latency).
Nmap done: 256 IP addresses (7 hosts up) scanned in 3.45 seconds
```

b) Sobre cada uno de los equipos que aparezcan como activos (excluido observador) realizar un escaneo de tipo *TCP connect scanning* [opción -sT] para determinar que puertos están abiertos.

```
nmap -sT -v 192.168.0.11
```

```
nmap -sT -v 192.168.0.22
```

c) Repetir el escaneado añadiendo la opción -O para que *nmap* trate de identificar el Sistema Operativo que ejecuta y la opción -sV para determinar la versión concreta de los servicios que tiene activados.

nmap -sT -O -sV 192.168.0.11

Los escaneados anteriores dejan rastro. Comprobar los ficheros de log (tail /var/log/syslog) en las máquinas y verificar si ha quedado constancia de las conexiones realizadas por *nmap*.

~# tail /var/log/syslog

Nota: El rastro del escaneo de tipo -sT que queda en /var/log/syslog fue guardado por el servidor telnet en el momento en que se estableció la conexión Telnet, por ello es necesario haber arrancado previamente el servidor Telnet (*/etc/init.d/openbsd-inetd start*).

Trabajo de Laboratorio 2

Capa de Transporte

Parte 1: TCP Fundamentos

1. Dos aplicaciones en las máquinas A y B intercambian datos utilizando el protocolo TCP.
 - a) Durante el intercambio de datos entre A y B, luego de establecida la conexión, la máquina A no recibe una confirmación de B sobre su ultimo envío, por lo que re-transmite el segmento no confirmado. Al cabo de un tiempo, B recibe las dos copias del segmento tal como A las envió. ¿B detecta que se trata de dos copias de la misma información? Justifique su respuesta.
 - b) Suponga que la conexión TCP que A inició con B, es una sesión telnet, (o sea que A se conectará al puerto 23 de B). ¿Es posible que A inicie una segunda conexión TCP hacia el puerto 23 de B, manteniendo la primera conexión activa? Justifique su respuesta e indique, en caso de ser posible, cómo diferenciará A y B los segmentos correspondientes a cada una de las conexiones.
2. Responda: Verdadero o Falso y justifique.
El uso de ventanas de un protocolo:
 - a) Permite aumentar la eficiencia de la transmisión.
 - b) Garantiza el control de errores.
 - c) Requiere la identificación de las tramas.
 - d) Disminuye el overhead del protocolo.
 - e) Realiza el control de CRC.
 - f) No acepta paquetes fuera de orden.
3. Utilice el software Cliente – Servidor de la parte 2 del TP1 y lance el analizador de protocolos para observar la siguiente secuencia:
 - a) Establezca conexión entre el cliente y el servidor, enviando un campo de datos del orden de 10 caracteres. Cierre luego la conexión.
 - b) Indique la cantidad de paquetes circularon entre mbos extremos desde el inicio de la conexión hasta su cierre.
 - c) Observe e indique las banderas que se observan en el 3-Way y 4-Way de inicio y cierre de la conexión.
 - d) Indique el número de secuencia que se observa en ambos extremos al inicio de la conexión. Y la ventana de Flujo de ambos extremos.
 - e) Indique los números de secuencia que se observan para ambos extremos al término del 3-Way state (sin envío de datos). Qué le parece particular?
 - f) Indique los números de secuencia que se observan para ambos extremos al inicio del cierre de la conexión.
 - g) Indique los números de secuencia que se observan para ambos extremos al término del cierre de la conexión. Qué le parece particular?

4. Dado el siguiente segmento TCP (se excluye la cabecera IP), enviado por un host A a otro B:

04 02 00 50 00 00 32 98 00 00 00 00 60 02 20 00
13 03 00 00 02 04 05 b4

- a) Qué función cumple el segmento enviado?
 - c) Caracterice a los Hosts A y B como cliente/servidor.
 - d) Cuál será el número de secuencia que utilizará el Host A en el próximo segmento que le envíe a B?
 - e) Cuál es el servicio requerido?
5. a. Suponga que la ventana de congestión de TCP está en 18 Kbytes. La ventana publicada por el otro extremo es de 64 Kbyte. ¿A qué valor llegará dicha ventana si los siguientes 5 segmentos transmitidos resultan exitosos y no se recibió aún ningún ACK? Suponga un tamaño máximo de segmento de 2 Kbytes.
- b. Suponga ahora que luego de transmitir los 5 segmentos exitosos llega un ACK acumulativo reconociendo tres paquetes y que el umbral se encuentra en 12Kbytes. A qué valor llegará la ventana de congestión.
6. Determine el tamaño óptimo de ventana para una sesión TCP en la que el RTT=100mseg, MSS=600 bytes y velocidad de la interfase 128 Kbps.
7. Qué mejora aporta el mecanismo de retransmisión rápida (fast retransmit) de TCP? Indique si aporta mejora, si la empeora o si es indiferente. Justifique.
- A. Permite que no llegue a ocurrir congestión en la red.
 - B. Permite que un proceso detecte que hay congestión antes de que esta alcance un valor demasiado alto.
 - C. Permite que la ventana de recepción pueda ser más grande.
 - D. Aumenta la Eficiencia de la comunicación.
 - E. Permite que la congestión ocurra antes.
8. Supongamos tener una situación en la cual se observa que al realizar una transferencia de archivo por FTP entre dos nodos, se observa que la transferencia efectiva de los datos es excesivamente lenta para el ancho de banda disponible. O sea que el ancho de banda consumido por la transferencia del archivo es muy pequeño si se compara con el ancho de banda total disponible entre los dos nodos. En cada uno de los casos siguientes, indique si la solución propuesta serviría para mejorar la situación y porqué (cómo afecta positiva o negativamente a la mejora de la velocidad de transferencia, o cómo no tiene ninguna influencia). Si los efectos de la solución dependen de otros parámetros no especificados, indíquese cuáles son estos.
- a. Se observa que la latencia de la red (retardo de transmisión) entre los dos procesos es muy alta. Como solución se propone aumentar el tamaño de la ventana de TCP en ambos nodos.

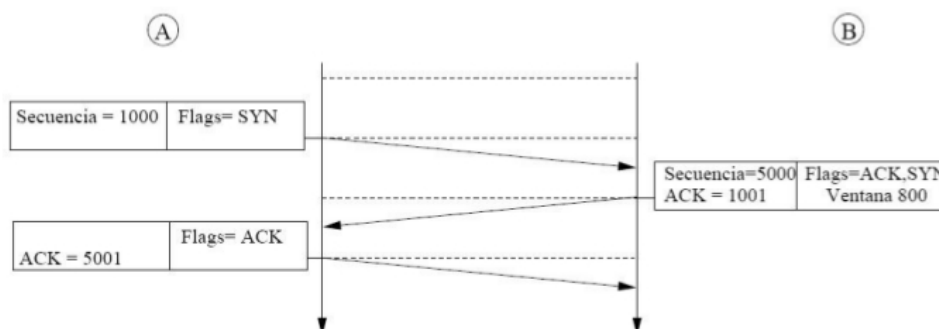
- b. Se observa que se están perdiendo muchos paquetes debido a errores en la red. Sin embargo, la latencia es muy baja. Como solución se propone disminuir el timeout de espera por asentimientos en los dos nodos.
- c. Se observa que se están reenviando muchos paquetes innecesariamente, porque los paquetes "originales" han llegado a destino. Como solución se propone aumentar el tamaño de la ventana de TCP en ambos nodos.
- d. Se observa que durante largos períodos cesa la emisión por llenarse la ventana de TCP del emisor. Como solución se propone disminuir el tamaño de la ventana del emisor y aumentar el plazo de espera por asentimientos.

Parte 2: TCP Análisis de Segmentos

9. De los escenarios **A. a F.**, en los cuales **las líneas punteadas representan tics de reloj**, resuelva sólo 2 o 3 de las situaciones planteadas, según se le inquiere. Año 2016: **A – C – E.**

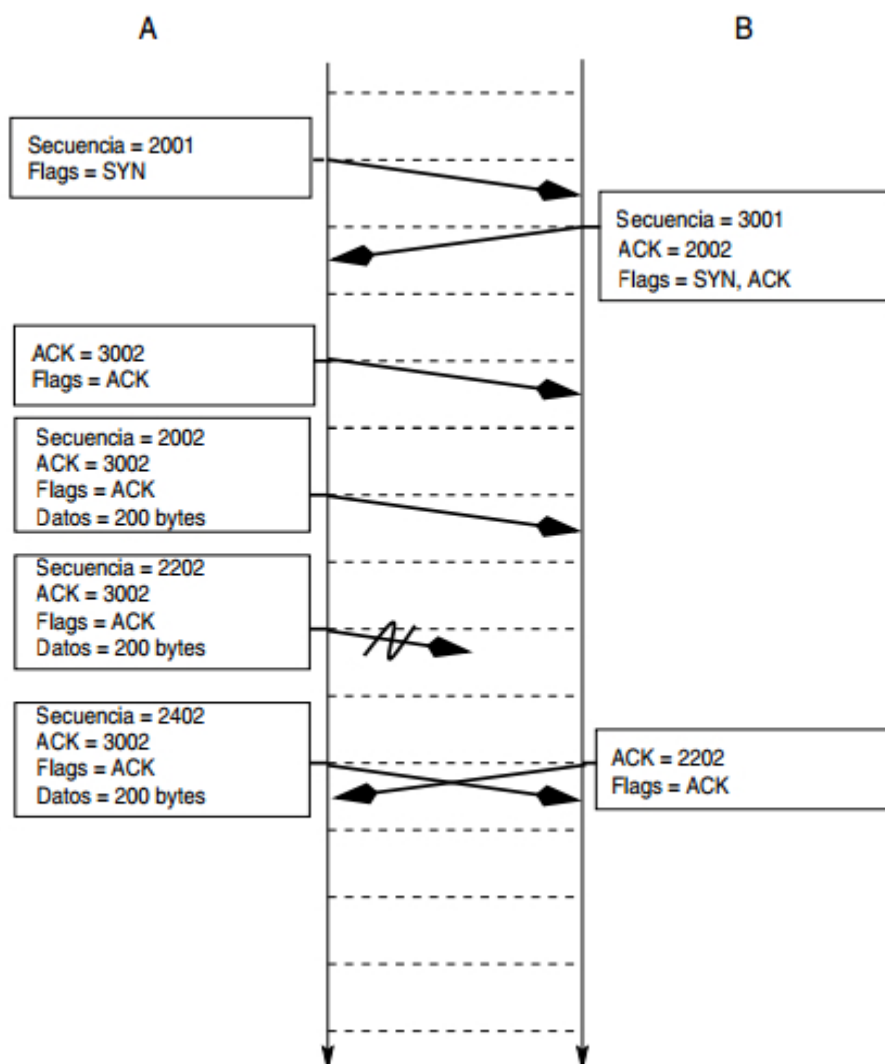
A. Complete la secuencia de envío de segmentos TCP reflejada en la figura, incluyendo el cierre de la conexión, sabiendo que:

- No se perderá ningún segmento en la transmisión excepto el cuarto con datos enviado por A.
- Los segmentos no dibujados (excepto el anteriormente citado) tardarán en llegar al destino hasta un tic de reloj y no se perderán.
- A está utilizando arranque lento (Slow Start) para prevenir la congestión.
- A tiene que enviar a B 800 bytes de datos, una vez enviados procederá cerrar la conexión.
- B no desea enviar datos a A.
- B enviará asentimientos a A cuando haya recibido dos segmentos de A desde el ultimo segmento asentido o cuando haya sucedido 1 tic de reloj desde desde el ultimo segmento recibido.
- El plazo de retransmisión de segmentos en A (timeout) es de 3 tics de reloj.
- A usa un tamaño fijo de datos de 200 bytes.
- B siempre enviará un valor de 800 en el campo de tamaño de la ventana de recepción.
- Tanto A como B sólo transmiten segmentos coincidiendo con el tic de reloj.
- A enviará segmentos con datos siempre que pueda.



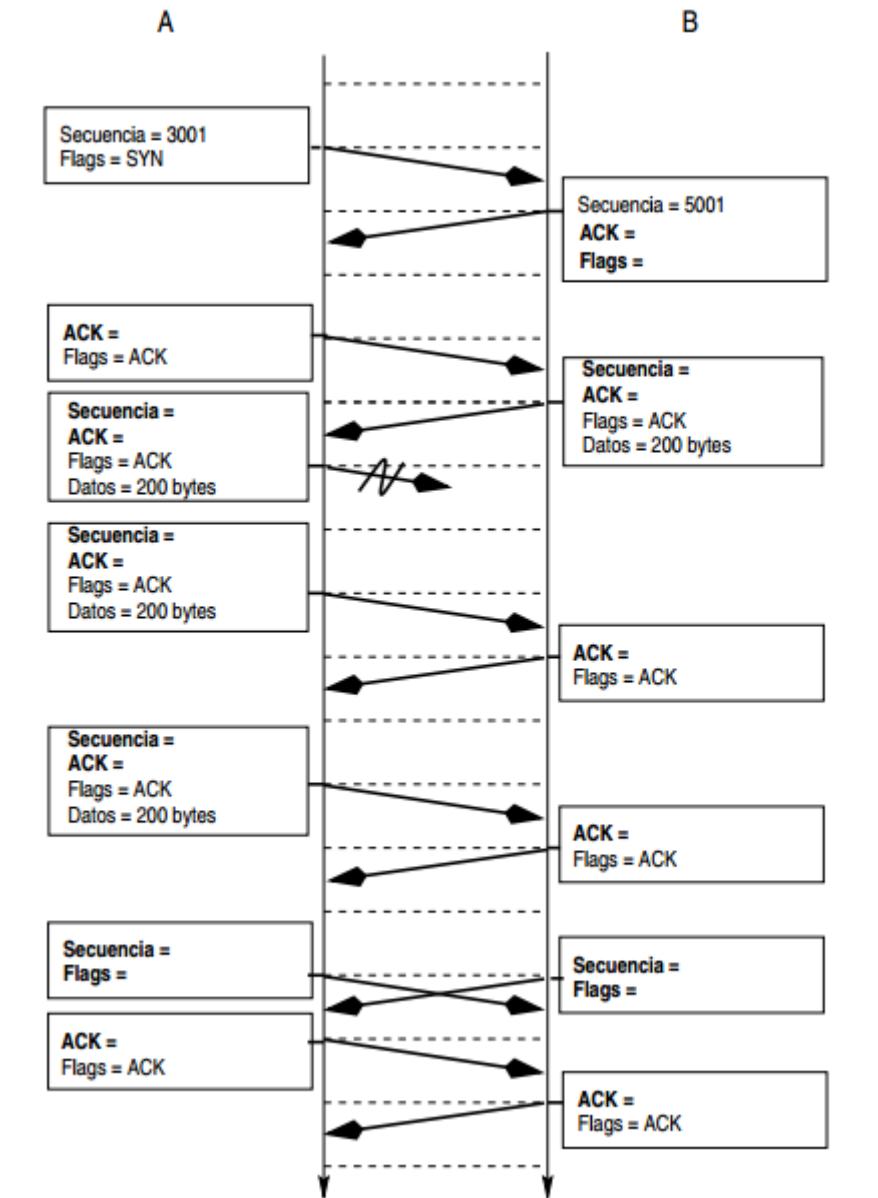
B. En la secuencia de envío de segmentos TCP reflejada en la figura, se sabe que ni A ni B quieren ya enviar más datos. Agregue los segmentos que faltan para que la conexión quede cerrada adecuadamente, suponiendo:

- Los segmentos que no se pierden, tardan en llegar al receptor hasta un tic de reloj.
- El plazo (Timeout) en que las máquinas esperan a que llegue un ACK es de 5 tics del reloj.
- A partir de lo ultimo dibujado en la figura, ya no se perderán más segmentos.
- B no haría nada hasta que le llegue otro segmento de A, momento en que responderá inmediatamente (en el siguiente tic de reloj) con un ACK.



C. En la figura se detalla la secuencia completa de envío de segmentos en una conexión TCP entre A y B (incluyendo apertura y cierre de la misma). Rellene los campos que faltan en la figura para que la misma tenga sentido, suponiendo que:

- El plazo en que las máquinas esperan a que llegue un ACK es de 5 tics del reloj.

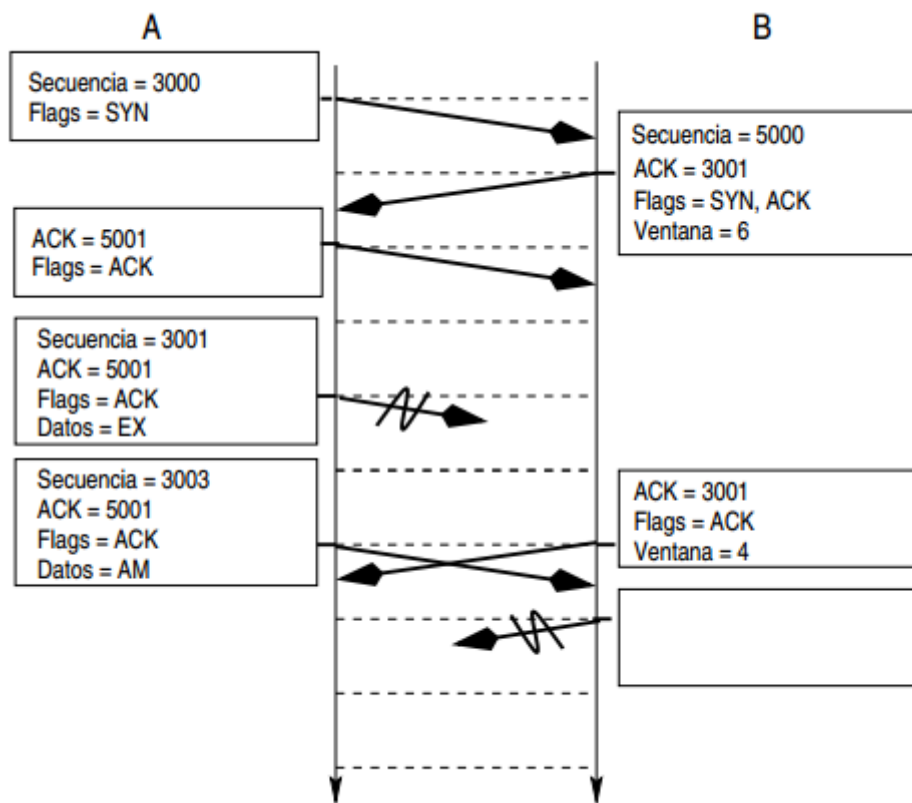


D. En la secuencia de envío de segmentos TCP reflejada en la figura, en la que las líneas horizontales representan tics de reloj, se sabe que:

- A desea enviar a B la cadena de caracteres "EXAMEN". B no tiene datos que enviar a A
- A usa un tamaño máximo de datos de 2 caracteres.
- Tanto A como B solo transmiten segmentos al principio del tic de reloj.
- Los segmentos que no se pierden, tardan en llegar al receptor hasta un tic de reloj.

- A tiene un plazo para retransmitir segmentos de 5 tics de reloj.
- A partir de los últimos segmentos dibujados en la figura:
 - o A enviará segmentos con datos siempre que pueda.
 - o B enviará un asentimiento cada vez que reciba un segmento de A, y ya no cambiará el tamaño de la ventana.
 - o Además del dibujado con el recuadro en blanco, el próximo segmento que envíe B también se perderá.
 - o No se perderá ningún otro segmento transmitido por A ni B.

Completar la transmisión en la figura (incluyendo el cierre de conexión) sin olvidar rellenar el recuadro en blanco.



E. En la secuencia de envío de segmentos TCP reflejada en la figura, en la que las líneas horizontales representan tics de reloj, se sabe que:

CASO 1:

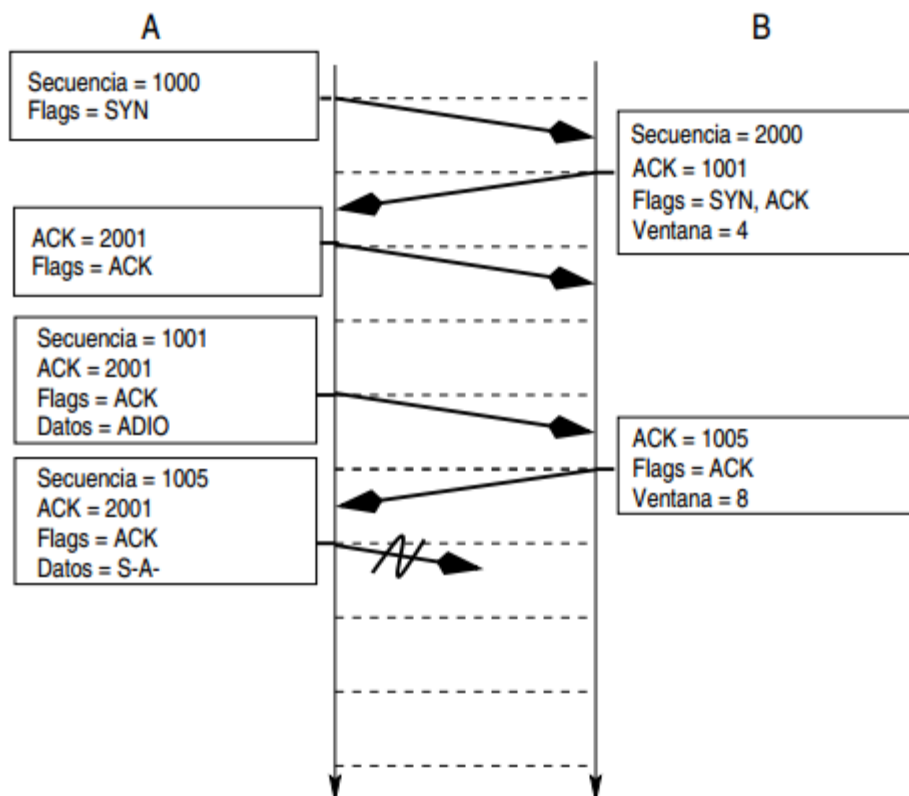
- A desea enviar a B la cadena de caracteres "ADIOS-A-TODOS". B no tiene datos que enviar a A.
- En la comunicación existe control de Flujo y Congestión.
- A usa un tamaño máximo de datos de 4 caracteres.
- Tanto A como B sólo transmiten segmentos al principio del tic de reloj.
- Los segmentos que no se pierden, tardan en llegar al receptor hasta un tic de reloj.

- Por Timeouts en el receptor, los reconocimientos pueden ser acumulativos de hasta dos paquetes por vez.
- A tiene un tiempo para retransmitir segmentos de 5 tics de reloj.
- A partir de los últimos segmentos dibujados en la figura:
 - o A enviará segmentos con datos siempre que pueda.
 - o B enviará un asentimiento inmediatamente cada vez que reciba un segmento de A y ya no cambiará el tamaño de la ventana.
 - o No se perderá ningún otro segmento transmitido por A ni B.

CASO 2:

- Los mismos requisitos que el Caso 1, sólo que A usa un tamaño máximo de datos de 2 caracteres y un Timeout de retransmisión de 7 Tics de reloj.

Completar la transmisión en la figura (incluyendo el cierre de conexión).

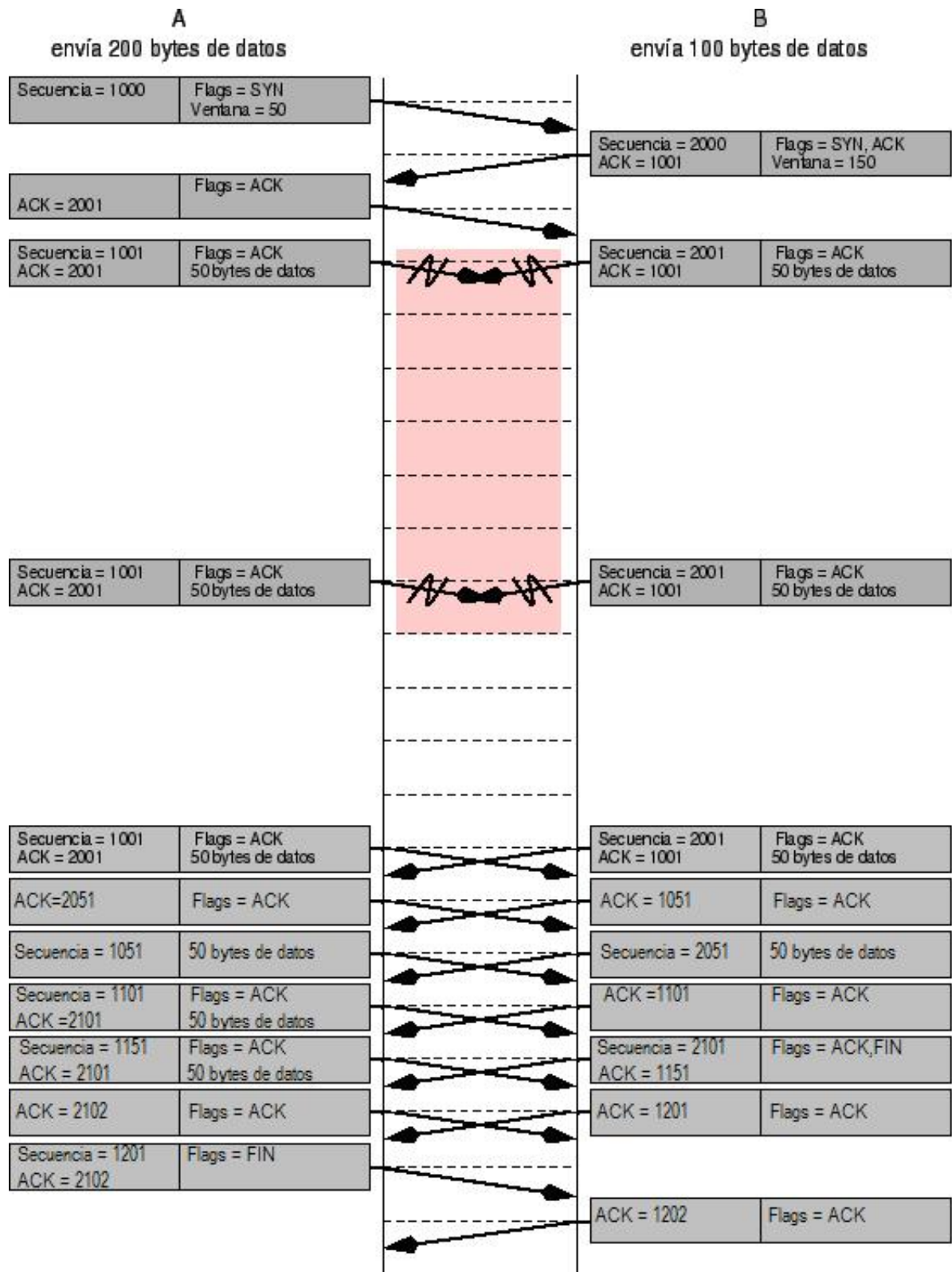


F. En la secuencia de envío de segmentos TCP reflejada en la figura, se sabe que:

- A desea enviar a B 200 bytes de datos. B desea enviar a A 100 bytes de datos.
- A y B usan un tamaño fijo de datos de 50 bytes.
- A y B ajustan la ventana acorde con "congestion avoidance".
- Tanto A como B sólo transmiten segmentos coincidiendo con el tic de reloj.
- Los segmentos que no se pierden, tardan en llegar al receptor hasta un tic de reloj.
- A y B tienen un plazo para retransmitir segmentos de 5 tics de reloj.

- A y B enviarán segmentos con datos siempre que puedan y enviarán un asentimiento cada vez que reciban un segmento con datos.

Teniendo en cuenta que la zona sombreada indica un periodo de tiempo durante el cual todos los segmentos transmitidos se perderán y que fuera de dicho período no se perderá ningún segmento, complete la transmisión en la figura (incluyendo el cierre de conexión).



Parte 3: TCP Análisis de Tramas

10. Análisis de capturas de tráfico de red e interpretación del mismo. Ahora vamos a estudiar e interpretar el segmento TCP que se encuentra dentro del paquete IP capturado.

A continuación una salida TCPDump de una conexión cualquiera.

```
08:46:01.903150 IP STATION.1472 > INFO01.3306: tcp 365 (DF)
0x0000  4500 0195 12ac 4000 8006 6341 c0a8 0122  E.....@...cA..."
0x0010  c0a8 0103 05c0 0cea 27dd 44a3 6fad 253b  ....'..D.o.%;
0x0020  5018 fd59 2def 0000 6901 0000 0355 5044  P..Y-...i....UPD
0x0030  4154 4520 7072 6a20 5345 5420 6465 7363  ATE.prj.SET.desc
0x0040  7269 703d 2027 4573 7465 2065 7320 756e  rip=.'Este.es.un
0x0050  2063 616d 706f 2074 6970 6f20 4d45 4d4f  .campo.tipo.MEMO
0x0060  2064 6f6e 6465 2073 6520 6775 6172 6461  .donde.se.guarda
0x0070  2069 6e66 6f72 6d61 6369 6f6e 2073 6f62  .informacion.sob
0x0080  7265 206c 6120 5c72 5c6e 6465 7363 7269  re.la.\r\ndescri
0x0090  7063 696f 6e20 6465 6c20 7072 6f79 6563  pcion.del.proyec
0x00a0  746f 2e20 5075 6564 6520 7365 7220 7465  to..Puede.ser.te
0x00b0  7874 6f20 6f20 696d 6167 656e 2e5c 725c  xto.o.imagen.\r\
0x00c0  6e53 6967 6f20 696e 7472 6f64 7563 6965  nSigo.introducie
0x00d0  6e64 6f20 6461 746f 7320 656e 206c 6120  ndo.datos.en.la.
0x00e0  6261 7365 2064 6520 6461 746f 7320 6d79  base.de.datos.my
0x00f0  7371 6c27 2057 4845 5245 2049 643d 3420  sql'.WHERE.Id=4.
0x0100  414e 4420 636c 6965 6e74 653d 2741 7374  AND.cliente='Ast
0x0110  696c 6c65 726f 7320 4573 7061 f16f 6c65  illeros.Espa.ole
0x0120  7320 532e 412e 2720 414e 4420 7072 6f79  s.S.A.'.AND.proy
0x0130  6563 746f 3d27 5072 6f79 6563 746f 2044  ecto='Proyecto.D
0x0140  6563 6f72 6163 69f3 6e20 3344 2070 6172  ecoraci.n.3D.par
0x0150  6120 6573 7061 6369 6f73 2042 756c 6b2d  a.espacios.Bulk-
0x0160  4361 7272 6965 7220 432f 3735 2720 414e  Carrier.C/75'.AN
0x0170  4420 61f1 6f3d 2731 3939 3527 2041 4e44  D.a.o='1995'.AND
0x0180  2075 726c 3d27 6363 335f 3164 3030 3030  .url='cc3_id0000
0x0190  2e6a 7067 27                                     be.jpg'
```

Debe distinguir las partes que a continuación se mencionan.

- a. Cabecera IP:
- b. Segmento TCP:
- c. Datos (incluye Opciones y Campo Relleno):
- d. Puerto origen:
- e. Puerto destino:
- f. Número de secuencia:
- g. Número de acuse de recibo:
- h. Posición de los datos (Data Offset):
- i. Campo reservado:
- j. Bits de código o indicadores. (6 bits):

- k. Window (ventana):
- l. Checksum o suma de verificación:
- m. Urgent Pointer o Puntero urgente:
- n. Opciones:
- o. Relleno:

11. Se realizó la captura de las siguientes tramas Ethernet. (tenga en cuenta que se extrajeron los bytes de preámbulo). Se pide: Analizar los campos relevantes de la información de **nivel de transporte** que contienen.

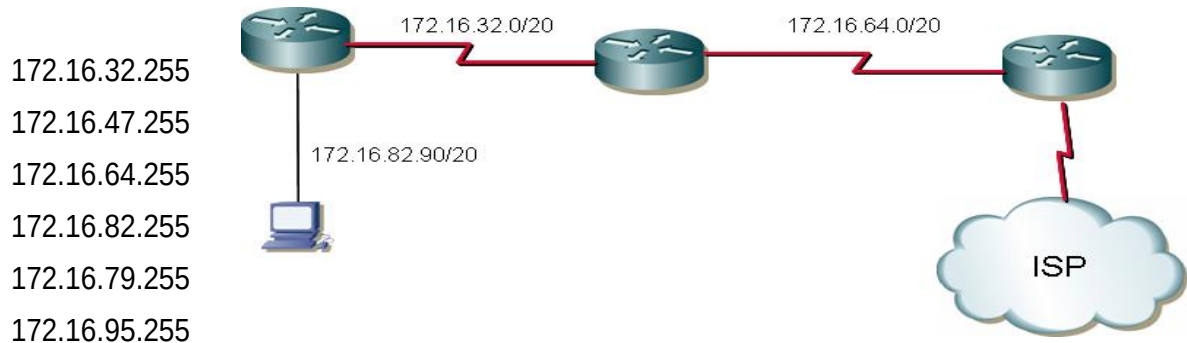
	Trama	Comentarios
1	00 18 f8 4e 70 2f 00 50 2c a4 34 ec 08 00 45 00 00 3e 7f 5e 00 00 80 11 cf aa c0 a8 01 64 c8 2a 61 6f 04 06 00 35 00 2a 2c a8 e4 e8 01 00 00 01 00 00 00 00 00 00 03 77 77 77 08 6d 69 6e 69 6e 6f 76 61 03 6f 72 67 00 00 01 00 01 23 cd ac f2	
2	00 50 2c a4 34 ec 00 18 f8 4e 70 2f 08 00 45 00 00 5c b7 fa 00 00 3c 11 da f0 c8 2a 61 6f c0 a8 01 64 00 35 04 06 00 48 36 15 e4 e8 81 80 00 01 00 02 00 00 00 00 03 77 77 77 08 6d 69 6e 69 6e 6f 76 61 03 6f 72 67 00 00 01 00 01 c0 0c 00 05 00 01 00 00 25 11 00 02 c0 10 c0 10 00 01 00 01 00 00 25 11 00 04 57 e9 93 8c a1 23 64 f3	
3	00 18 f8 4e 70 2f 00 50 2c a4 34 ec 08 00 45 00 00 30 7f 61 40 00 80 06 cd e4 c0 a8 01 64 57 e9 93 8c 0c 54 00 50 f0 e8 a3 97 00 00 00 00 70 02 ff ff 34 79 00 00 02 04 05 b4 01 01 04 02 11 ac fb 4b	
4	00 50 2c a4 34 ec 00 18 f8 4e 70 2f 08 00 45 00 00 30 00 00 40 00 35 06 98 46 57 e9 93 8c c0 a8 01 64 00 50 0c 54 16 3b ae 0d f0 e8 a3 98 70 12 6 d0 59 4f 00 00 02 04 05 b4 01 01 04 02 ac 23 23 ca	
5	00 18 f8 4e 70 2f 00 50 2c a4 34 ec 08 00 45 00 00 28 7f 62 40 00 80 06 cd eb c0 a8 01 64 57 e9 93 8c 0c 54 00 50 f0 e8 a3 98 16 3b ae 0e 50 10 ff ff 9c e3 00 00 a2 cb 23 45	
6	00 18 f8 4e 70 2f 00 50 2c a4 34 ec 08 00 45 00 01 c0 7f 63 40 00 80 06 cc 52 c0 a8 01 64 57 e9 93 8c 0c 54 00 50 f0 e8 a3 98 16 3b ae 0e 50 18 ff ff ee 95 00 00 47 45 54 20 2f 20 48 54 54 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 77 77 77 2e 6d 69 6e 69 6e 6f 76 61 2e 6f 72 67 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 35 2e 30 20 28 57 69 6e 64 6f 77 73 3b 20 55 3b 20 57 69 6e 64 6f 77 73 20 4e 54 20 35 2e	

Trabajo de Laboratorio 3

Capa de Red

Parte 1: IP Fundamentos -Subnetting

1. En el gráfico se indica la red del rectorado de la UNP. Elija entre las opciones dadas las que correspondan a direcciones broadcast.



2. Ud. está configurando una subred de la oficina central de los supermercados “La Ocasión”. Necesita asignar direcciones IP de hosts de la subred. Definió la máscara 255.255.255.224.

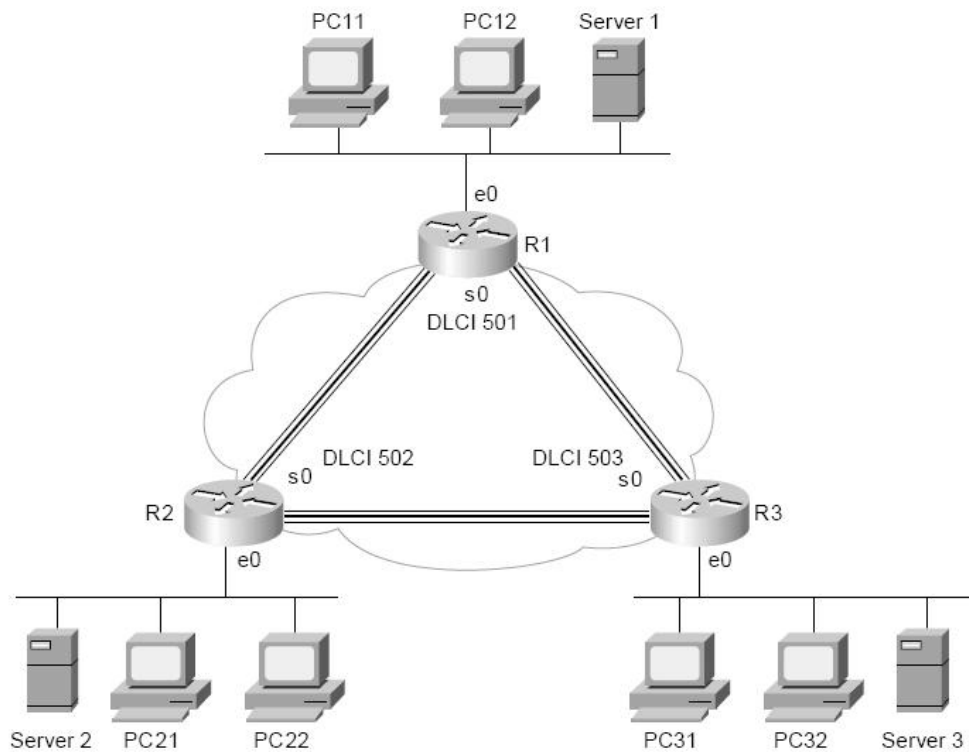
a. Cuáles de las siguientes direcciones son válidas para asignar a host?.

- 1) 15.234.118.63
- 2) 92.11.178.93
- 3) 134.178.18.56
- 4) 192.168.16.87
- 5) 201.45.116.159
- 6) 217.63.12.192

b. Ud. debe particionar la red 172.12.0.0 en subredes tales que puedan alojar 458 hosts IP. Además se debe lograr el máximo número de subredes. Determine la máscara a utilizar.

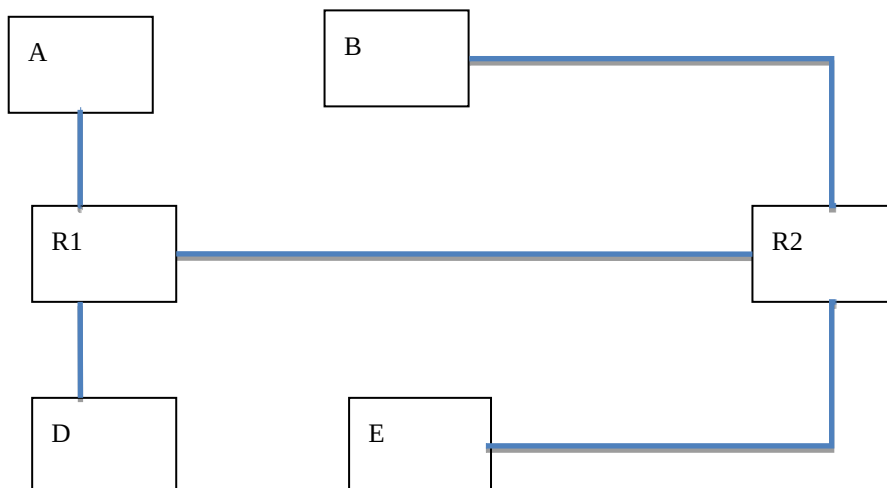
3. Para el esquema de la figura complete la siguiente tabla. Considere que se ha utilizado la máscara 255.255.255.192

Interfaz Router	Dirección IP	Dirección Subred	Rango de direcciones válidas
R1E0	168.11.11.101		
R2E0	168.11.12.102		
R3E0	168.11.13.103		
R1S0	168.11.123.201		
R2S0	168.11.123.202		
R3S0	168.11.123.203		



4. El siguiente es un sistema con subredes A,B,D y E y 2 routers R1 y R2. Se desea hacer subnetting utilizando la dirección que nos otorga el ISP 200.37.6.128/28. Indicar la dirección de cada subred que es necesario asignar teniendo en cuenta que:
- A debe contener 7 hosts.
 - B debe contener 15 hosts.
 - D debe contener 30 hosts.
 - E debe contener 13 hosts.

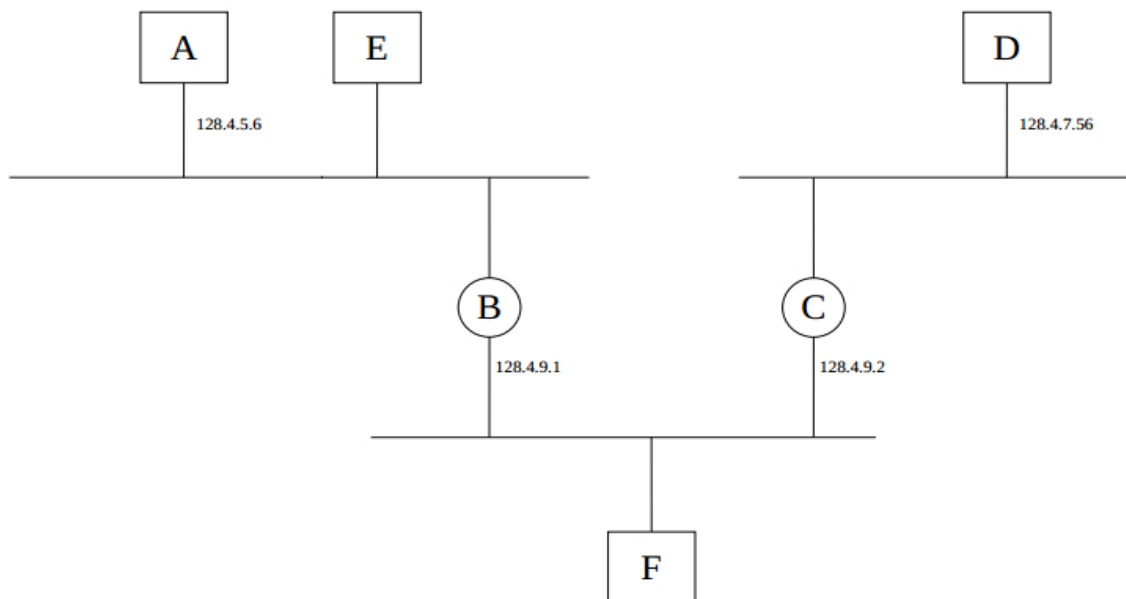
En caso que la dirección otorgada por el ISP no alcance para efectuar el subnetting deseado, cual sería la máscara óptima que se requiere.



Parte 2: IP Fragmentación – ICMP - Encaminamiento

5. Un router recibe un datagrama con header básico y longitud total **832 bytes**. Debe ser enviado a un segmento en el que el MTU es de **520 bytes**. Determine en forma completa, acorde con los datos suministrados, los campos principales de los encabezados de los segmentos que se generarán.
6. Dada la situación representada en la figura:
- Asignar razonablemente direcciones IP válidas a las interfaces de red a las que les falte.
 - Establecer tablas de encaminamiento para que (simultáneamente):
 - A hable con D y viceversa
 - E hable con C pero no con D
 - A no pueda hablar con F

NOTA: La máscara de subred es 255.255.255.0 en todos los casos.

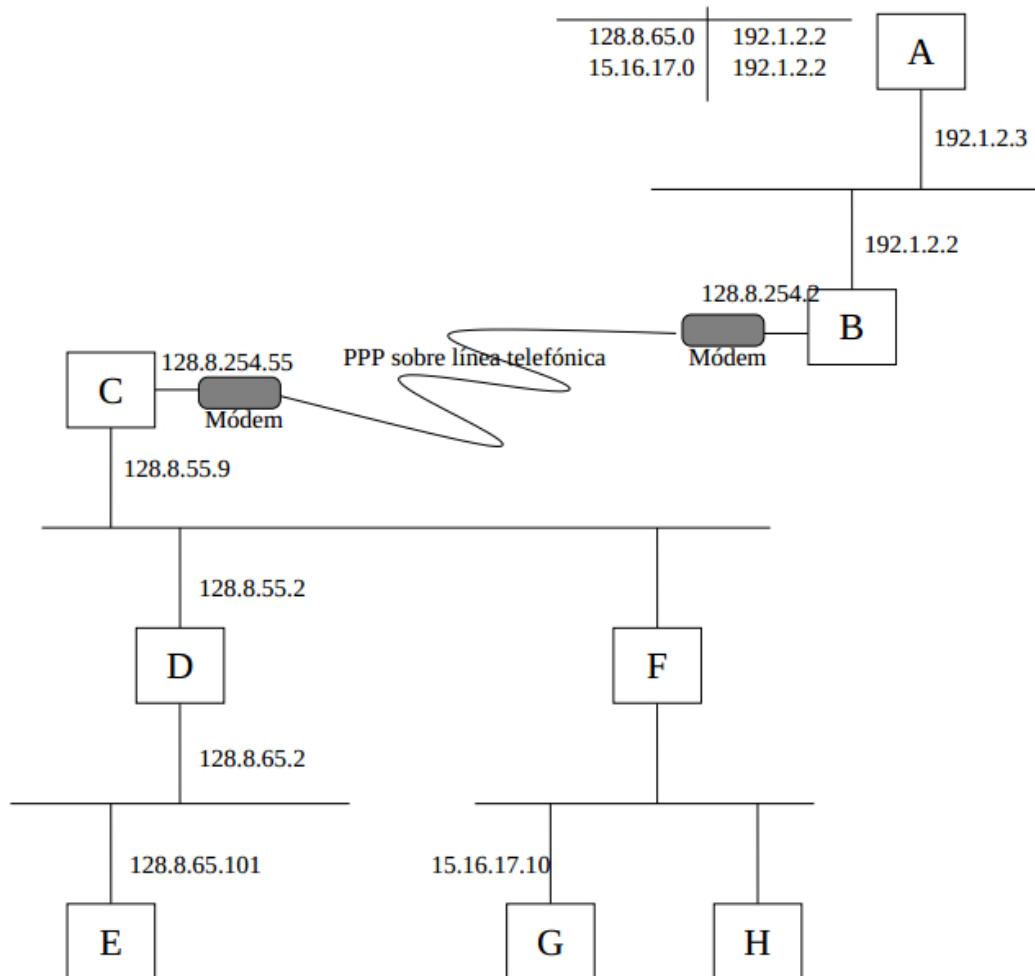


7. En la figura se muestra una red basada en protocolos TCP/IP. La máscara de cada subred es 255.255.255.0. No se permiten rutas por defecto.

La tabla de encaminamiento de A se muestra en la figura. El resto de tablas se supone que **no impiden** ningún camino de comunicación.

Se pide:

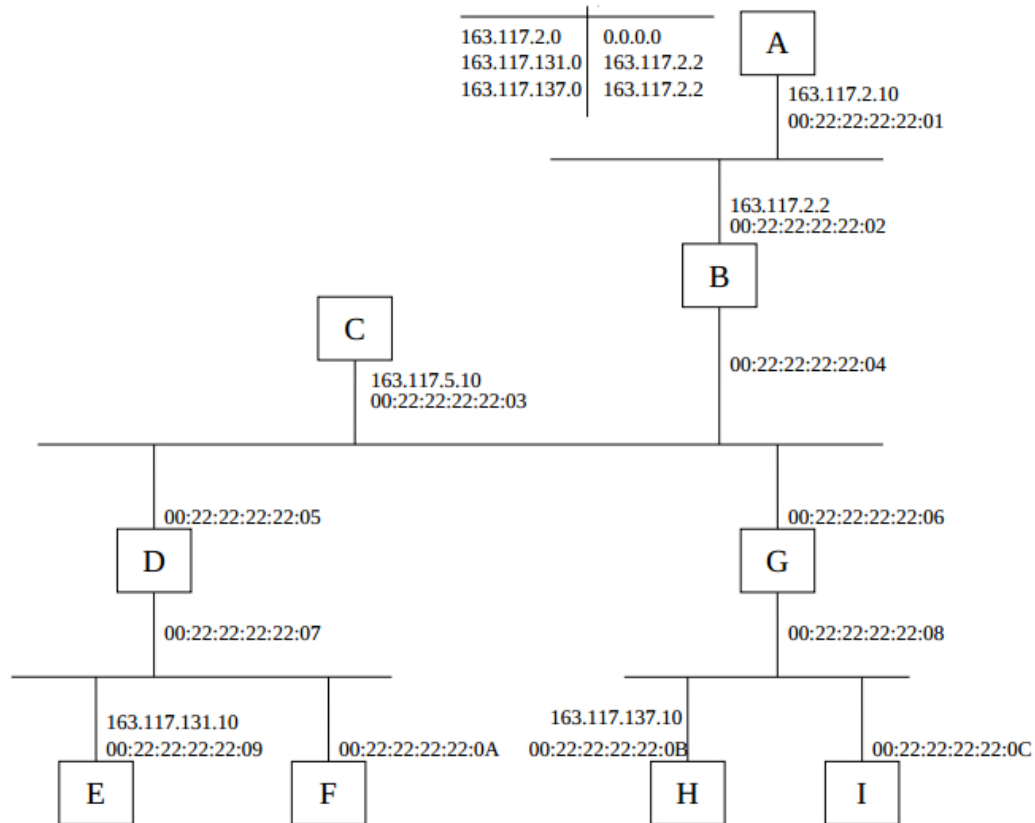
- Qué dirección IP podría tener H? ¿Y F? Justifique.
- Puede hablar A con C? Justifique.
- Puede hablar A con E? Justifique.
- Cómo habría que actualizar alguna tabla de encaminamiento para que A hable con C pero no con D? Justifique.



8. En la figura siguiente se muestra una red basada en protocolos TCP/IP. En cada host se indica su dirección de capa de enlace (MAC) y en algunos casos, su dirección IP de la red a la que pertenece. La máscara de todas las subredes es /24. La tabla de encaminamiento de A se muestra en la figura. Se supone que el resto de las tablas de encaminamiento no impiden ningún camino de comunicación.

Se pide:

- Asignar razonablemente todas las direcciones IP que faltan en la figura.
- Puede hablar A con C?. Justifique.
- Puede hablar A con E?. Justifique.
- Cómo habría que modificar la tabla de A para que pueda hablar con E pero no con F?. Justifique.



9. En el esquema de la figura las rutas de Piglet se sustituyen de :

Piglet(config)# ip route 192.168.1.0 255.255.255.0 192.168.1.193

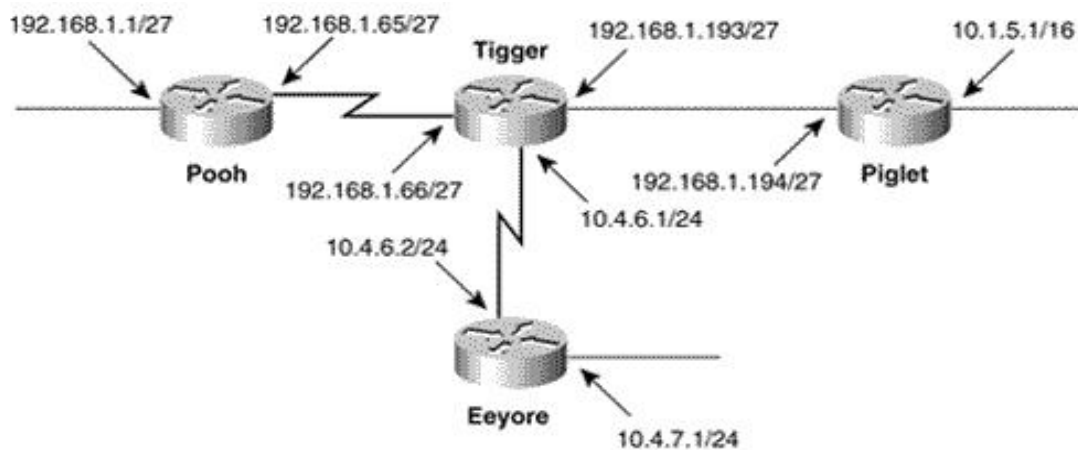
Piglet(config)# ip route 10.4.0.0 255.255.0.0 192.168.1.193

por estas otras:

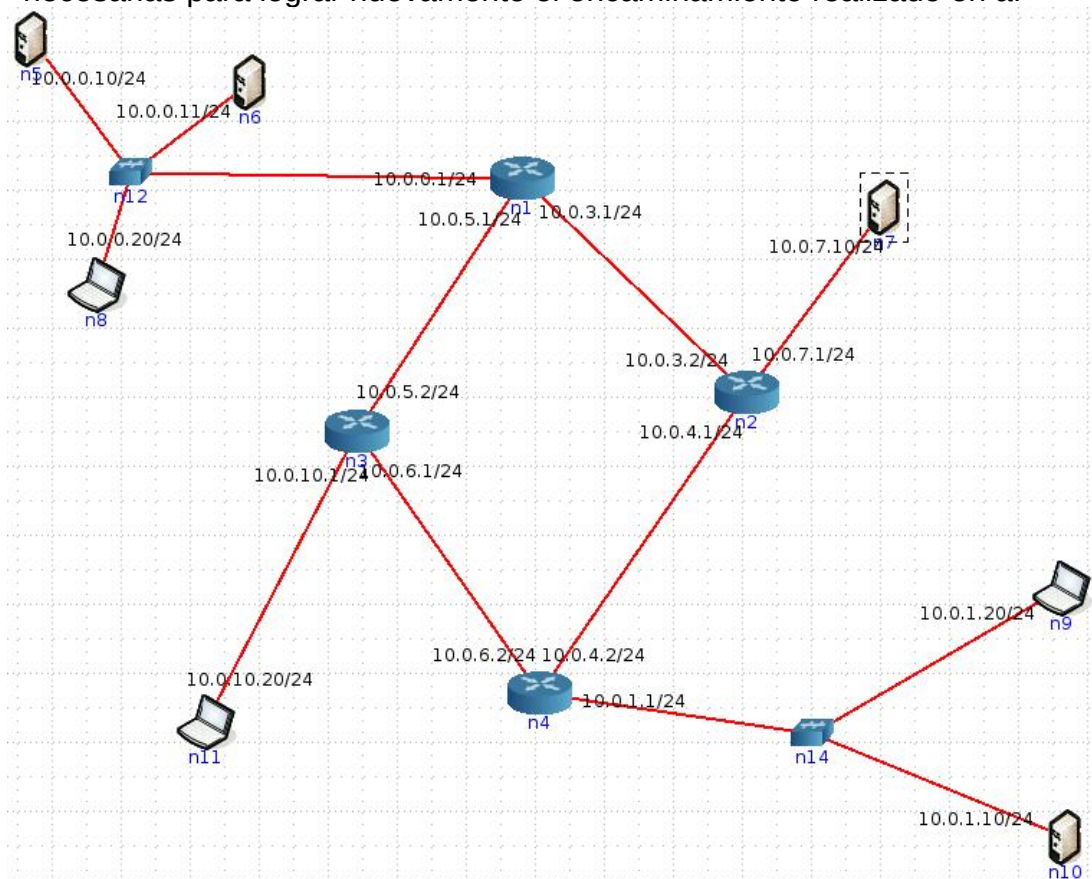
Piglet(config)# ip route 192.168.1.0 255.255.255.224 192.168.1.193

Piglet(config)# ip route 10.0.0.0 255.255.0.0 192.168.1.193

Cuál es el resultado? **Nota: Tenga en cuenta que las reglas para sus redes adyacentes no se eliminan.**



10. Desarrollar con la herramienta CORE la red de la figura. Se podrán manejar las direcciones IP que automáticamente asigne el programa. En estas condiciones efectuar los siguientes pasos:
- Utilizar el comando `ifconfig` para averiguar las direcciones IP otorgadas a los routers.
 - Efectuar un Ping desde la máquina n6 a n10. Consignar RTT.
 - Observar las tablas de encaminamiento del router n1 a efectos de ver por cuál camino se enruta el mensaje anterior. Utilice el comando `traceroute` desde el icono 1—2 del aplicativo para dibujar la ruta en el diagrama.
 - Con el comando `iptables` neutralice el forwarding que se efectúa en el router n2 o n3 que encamina hacia n10. Observe el resultado.
 - Agregue en el router n1 y en el opuesto (n2 o n3) en diagonal al anterior las rutas necesarias para lograr nuevamente el encaminamiento realizado en a.

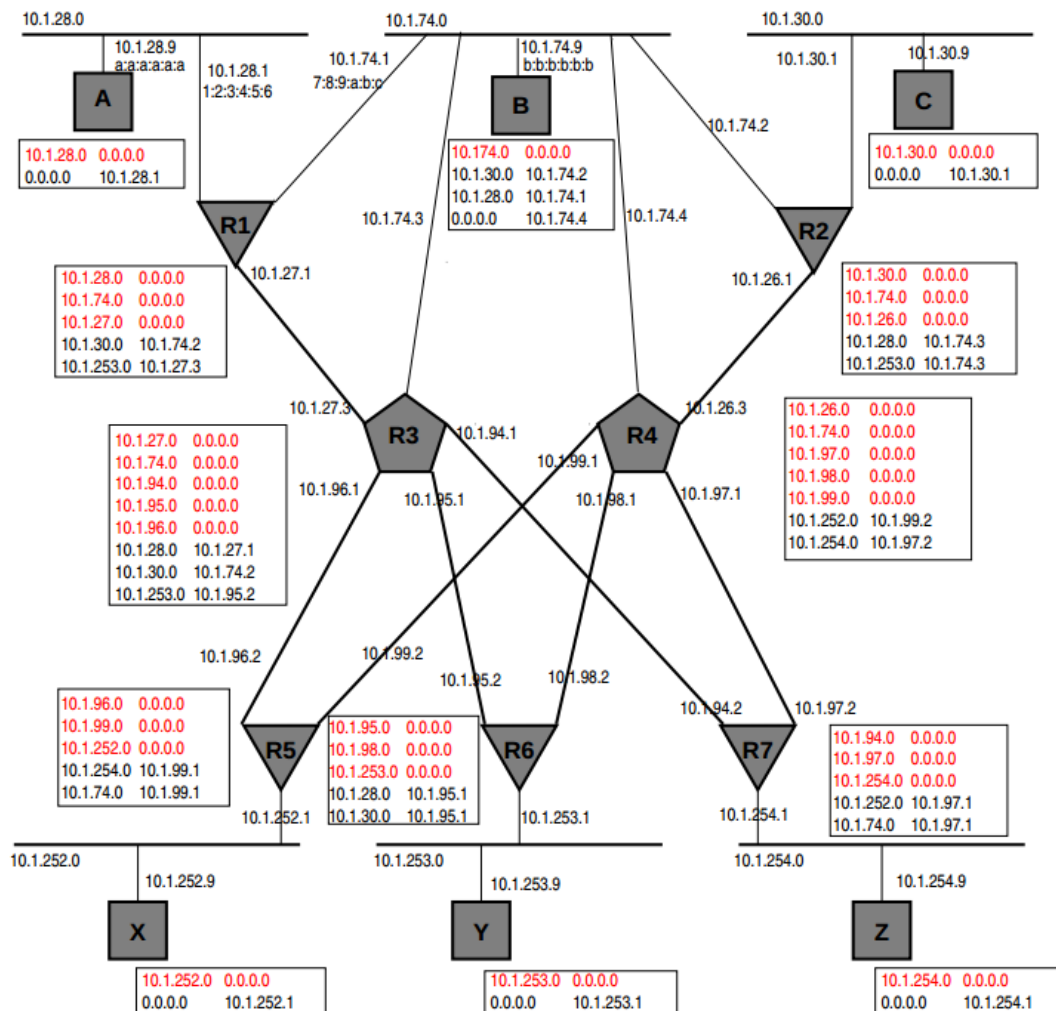


Parte 3: Ip Routing - NAT

11. Dada la red de la figura siguiente, donde la máscara de subred es /24 y considerando la tabla de encaminamiento que aparece más abajo, se solicita:
- Completar la tabla indicando en la casillas en blanco:
 - NO** cuando no hay camino desde M hasta N.
 - La secuencia de encaminadores intermedios cuando sí hay camino desde M hasta N.

	A	B	C	X	Y	Z
A	--	R1		NO	R1,R3,R6	
B	R1	--	R2		NO	

C		R2	--			NO
X	NO			--	NO	
Y	R6,R3,R1	NO		NO	--	
Z			NO			--



- Supongamos que se cae el encaminador R3, quedando fuera de servicio. Reconstruir las tablas de R1, R2, R4, R5, R6 y R7 para que sigan pudiendo comunicarse entre sí las mismas máquinas que lo hacían antes.
- Adaptar las tablas de los encaminadores (routers) para que pueda circular todo el tráfico con origen en X y destino en Z, pero sólo de forma que pase por la subred de B. Indicar sólo las rutas que habría que quitar, añadir o modificar. Se valorará realizar el menor numero de cambios en las tablas

12. En una red cuya topología se indica en la figura, la máquina 10.0.2.10 envía un paquete IP a la máquina 10.0.0.10.

- Indique el camino seguido por el paquete, detallando cada enrutador que atraviesa y la decisión que se toma en el mismo.
- Indique si observa alguna anomalía y de existir cuál sería su solución.
- Haga el mismo análisis que en a. y b. Pero cambiando la máscara de la red a 10.0.0.0/24.

Direcciones IP de los routers

Router A		Router B	
10.0.0.1	Interfaz1	10.0.0.134	Interfaz1
10.0.0.129	Interfaz2	10.0.0.137	Interfaz2
10.0.0.133	Interfaz3	10.0.0.145	Interfaz3
		10.0.0.149	Interfaz4

Router C		Router D	
10.0.0.141	Interfaz1	10.0.0.142	Interfaz1
10.0.0.138	Interfaz2	10.0.0.146	Interfaz2
10.0.0.130	Interfaz3	10.0.0.153	Interfaz3

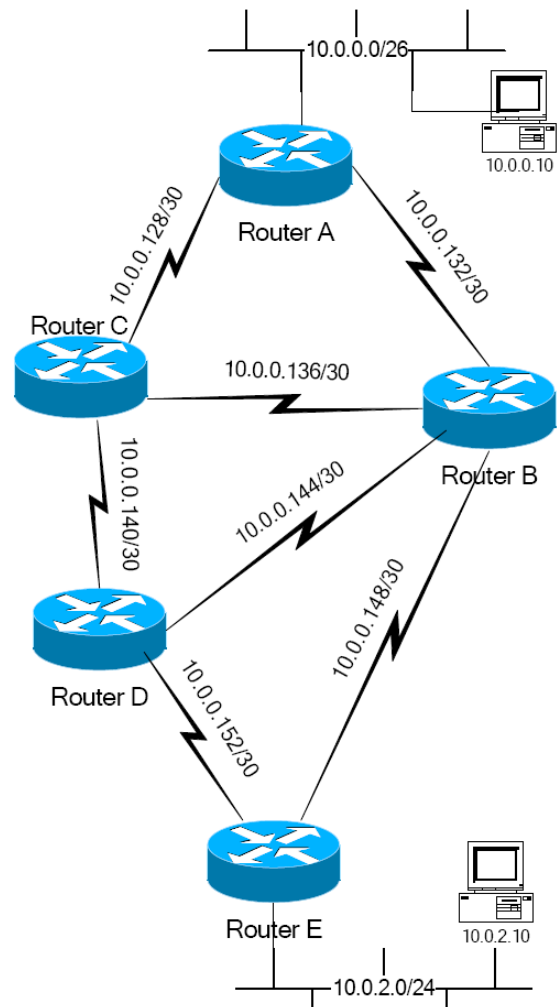
Router E	
10.0.2.1	Interfaz1
10.0.0.154	Interfaz2
10.0.0.150	Interfaz3

Información de Ruteo

Router A		Router B	
Destino	Proximo Salto	Destino	Proximo Salto
10.0.0.0/26	conectada	10.0.0.132/30	conectada
10.0.0.128/30	conectada	10.0.0.136/30	conectada
10.0.0.132/30	conectada	10.0.0.144/30	conectada
10.0.2.0/24	10.0.0.134	10.0.0.148/30	conectada
		10.0.0.0/24	10.0.0.138
		10.0.0.0/23	10.0.0.133
		10.0.2.0/24	10.0.0.150

Router C		Router D	
Destino	Proximo Salto	Destino	Proximo Salto
10.0.0.140/30	conectada	10.0.0.140/30	conectada
10.0.0.136/30	conectada	10.0.0.144/30	conectada
10.0.0.128/30	conectada	10.0.0.152/30	conectada
10.0.0.0/24	10.0.0.129	10.0.0.0/24	10.0.0.141
10.0.0.0/25	10.0.0.142	10.0.0.0/25	10.0.0.145

Router E	
Destino	Proximo Salto
10.0.2.0/24	conectada
10.0.0.152/30	conectada
10.0.0.148/30	conectada
10.0.0.0/24	10.0.0.153
10.0.0.0/25	10.0.0.149



Nota: Las PC tienen una ruta por defecto al enrutador conectado

13. En la red de la figura anterior, al router B se le agrega una placa Ethernet y se conecta a Internet con un esquema de NAT.

- a. Indique una dirección válida que pueda otorgarle el ISP.
- b. Si los dos HOST que se indican en la figura establecen una conexión cada uno, simultáneas con el servidor WEB de la Universidad de la patagonia cuya dirección IP es 200.37.12.88 y a su vez otra conexión cada uno de los HOST con el Servidor de Correo SMTP de Google (IP= 12.18.32.25), indique cuáles serán las direcciones y puertos que se utilizarán a un lado y otro del Router B, en uno y otro sentido, para permitir las cuatro conexiones apuntadas
- c. Agregue las rutas necesarias a los routers asociados a los hosts en cuestión a efectos que el enrutamiento a internet sea posible sin restricciones.

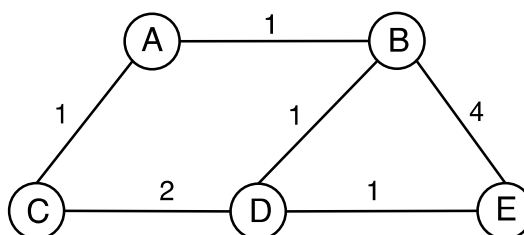
14.Cuál de las siguientes afirmaciones son correctas:

Al encapsularse los mensajes de RIP en datagramas UDP:

- a. RIP no sufre de problemas de la fragmentación de datagramas IP
- b. RIP no sufre los problemas de descartado de paquetes por congestión en encaminadores routers.
- c. Los mensajes de RIP se desencapsulan por número de puerto del datagrama UDP.
- d. Los mensajes de RIP se desencapsulan por número de protocolo en el datagrama IP

15. La figura muestra una red en la que se utiliza un protocolo de encaminamiento del tipo de estado de enlace. Las cifras sobre los enlaces indican el valor de distancia entre nodos en un instante dado.

- a. Indica la información que contendrá el próximo paquete de estado de enlace que enviará el nodo B. ¿A qué nodos llegará esta información?
- b. Supongamos que ahora se cambia el protocolo de encaminamiento por uno de vector de distancias. Indica la información que enviará el nodo B. ¿A qué nodos llegará esta información?
- c. Atendiendo únicamente al **número total de mensajes** de información de encaminamiento que se generan en una ronda, explica razonadamente si para la red de la figura es mejor usar un algoritmo de encaminamiento de estado de enlace o de vector de distancias.
- d. Atendiendo únicamente al **tamaño de los mensajes** de información de encaminamiento que se generan en una ronda, explica razonadamente si para la red de la figura es mejor usar un algoritmo de encaminamiento de estado de enlace o de vector de distancias.



16. Juan está utilizando la computadora de su casa y no puede comunicarse con la máquina de su universidad (gsync.escet.urjc.es).

Al hacer un ping obtiene el siguiente mensaje:

```
vmo$ ping gsync.escet.urjc.es
PING gsync.escet.urjc.es (193.147.71.64): 56 data bytes
36 bytes from v100.mpd01.mad05.atlas.cogentco.com (130.117.1.38): Time to live
exceeded
Vr HL TOS Len ID Flg off TTL Pro cks Src Dst4 5 00 0054 2b7b 0 0000 01 01 c3b0
192.168.1.2 193.147.71.64
```

Al hacer un traceroute obtiene el siguiente mensaje:

```
vmo$ traceroute gsync.escet.urjc.es
traceroute to gsync.escet.urjc.es (193.147.71.64), 64 hops max, 40 byte packets
1 mygateway.ar7 (192.168.1.1) 48.066 ms 3.932 ms 3.173 ms
2 madhou.jazztel.es (212.106.217.74) 45.129 ms 68.204 ms 43.838 ms
3 madhou.jazztel.es (212.106.217.4) 44.337 ms 44.953 ms 44.815 ms
4 t3-4.mpd01.mad05.atlas.cogentco.com (130.117.241.221) 45.579 ms 70.780 ms
44.525 ms
5 ***
6 v100.mpd01.mad05.atlas.cogentco.com (130.117.1.38) 63.228 ms 44.523 ms
44.619 ms
7 ***
8 v100.mpd01.mad05.atlas.cogentco.com (130.117.1.38) 59.143 ms 48.010 ms
45.734 ms
9 ***
```

... ..

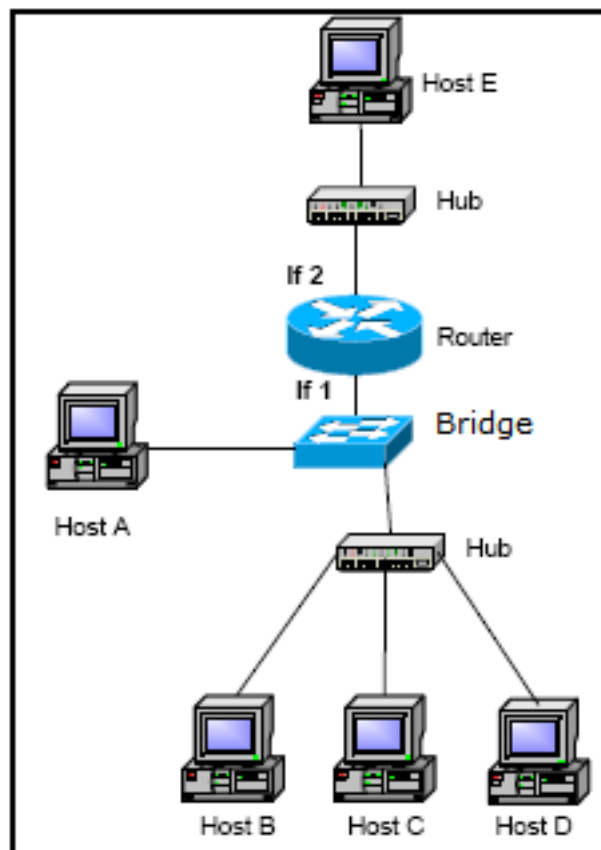
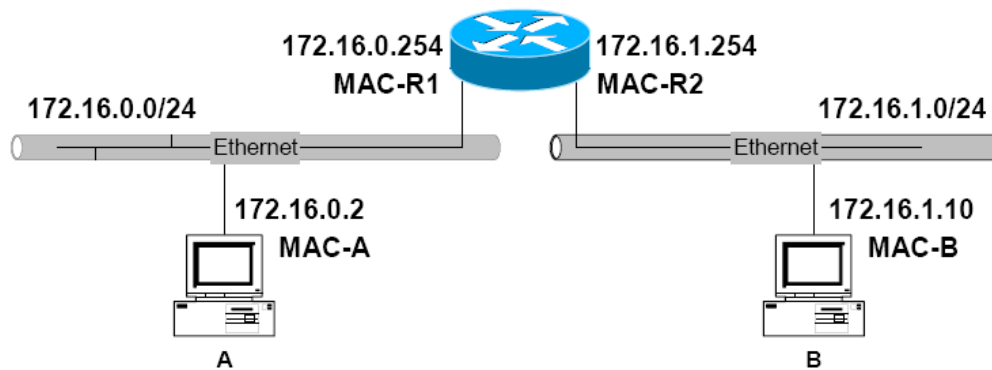
- a. Cuál cree que es el problema? En qué nivel de las capas OSI?
- b. Cuál es la dirección IP de la máquina de Juan y porqué? Dibuje un esquema de las máquinas que aparecen en las trazas.
- c. Razoné acerca de qué tipo de acceso a Internet cree que tiene Juan.
- d. Una máquina con la dirección IP= 192.168.1.3, en qué red o redes del esquema anterior podría estar? y una con la IP= 193.147.71.110?
- e. Escriba la entrada de la tabla de encaminamiento de la máquina mygateway.ar7 que se usa cuando se ejecutan el ping y traceroute anteriores.
- f. A la vista de estas trazas, cree que Juan se podría conectar a pantuflo.dat.escet.urjc.es? Si es que sí en que supuestos y si es que no justifíquelo. Lo mismo para mimaquina.alojamientos.com.
- g. Suponiendo que las cachés de ARP están vacías antes del primer ping y con el esquema dibujado en el punto b, indicar los campos más relevantes de los paquetes ARP que se habrán generado como consecuencia del ping en las subredes de origen y destino.

Trabajo de Laboratorio 4

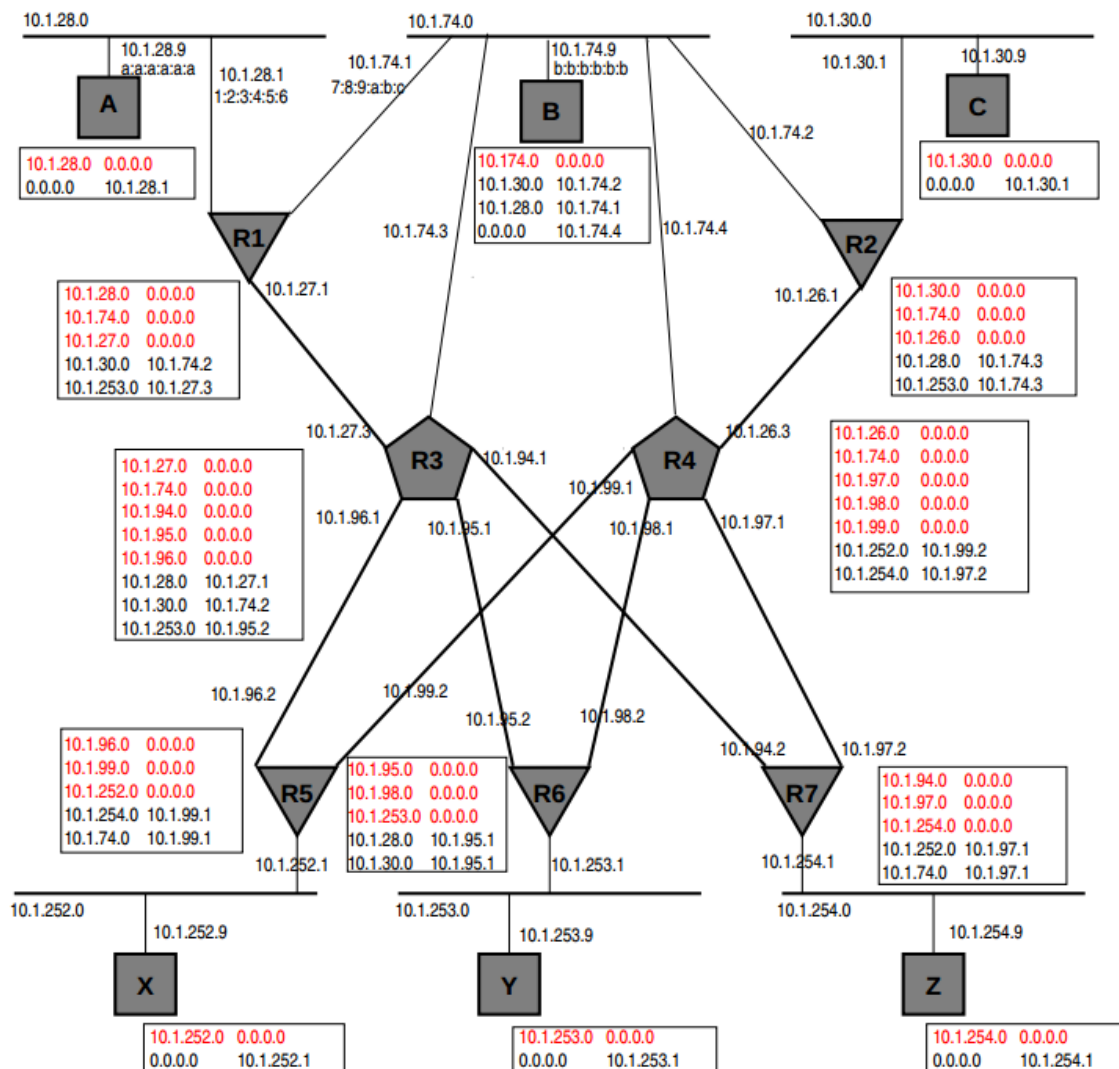
Capa de Enlace y Física

Parte 1: Enlace - MAC – ARP

1. En el diagrama de la figura, el usuario en la máquina A ejecuta “ping 172.16.1.10”. En el paquete ICMP dirigido desde A hacia B, visto en la LAN donde está conectada A, indique cuáles son las direcciones de capa MAC origen y destino, y las direcciones IP origen y destino. Indique cómo decidió la máquina A utilizar estas direcciones. Suponga que todos los equipos conocen las direcciones MAC que necesitan.



2. Dada la topología de la figura anterior, responder las siguientes preguntas. Justifique cada una de sus respuestas.
- Es posible que colisionen dos tramas, una desde Host A a Host C y otra desde Host B a Host D?
 - Cuales MAC conoce o puede llegar a conocer el Host B?
 - El Host C realiza un pedido de ARP (Broadcast de capa2), liste los Host que logran ver el pedido.
3. Dada la topología de la figura siguiente, responder las siguientes preguntas. Justifique cada una de sus respuestas.



- Indicar si la siguiente trama Ethernet puede ser una trama correcta generada en la red de la figura (explicando cuándo y dónde se genera) o no:

Eth. Destino	Eth. Origen	Protocolo	IP Origen	IP Destino	...
b:b:b:b:b:b	a:a:a:a:a:a	IP	10.1.28.9	10.1.74.9	...

- b. Indicar qué tipo de trama es la siguiente, para qué se generaría en la red de la figura y cuáles serían los campos relevantes que le faltan:

Eth. Destino	Eth. Origen	Protocolo	...
1:2:3:4:5:6	a:a:a:a:a:a	ARP	...

- c. Indicar la salida en pantalla que genera el comando traceroute que hay que invocar para que aparezca en la red de la figura la siguiente trama:

Eth. Destino	Eth. Origen	Protocolo	IP Origen	IP Destino	Protocolo	...
b:b:b:b:b:b	7:8:9:a:b:c	IP	10.1.28.9	10.1.74.9	UDP	...

4. Utilizando la red de la figura anterior, teniendo en cuenta que la máscara de subred es siempre /24, se solicita:
- Sea un paquete IP con origen en A y destino en B. Si miramos los contenidos de la trama Ethernet en que va encapsulado cuando pasa por la red 10.1.28.0, y los comparamos con los de la trama Ethernet de cuando pasa por la 10.1.74.0, indicar qué campos serán diferentes, tanto de la trama Ethernet como del paquete IP. Señalar, cuando se pueda, el valor que tendría cada uno de esos campos en ambas tramas.
 - Con los datos de la figura, indicar en qué subredes Ethernet puede encontrarse un paquete IP con dirección destino 10.1.28.9.
 - Con los datos de la figura, indicar en qué subredes Ethernet puede encontrarse un paquete IP con dirección origen 10.1.252.9.
 - Con los datos de la figura, indicar en qué subredes Ethernet puede encontrarse un paquete IP con dirección origen 10.1.254.9 y dirección destino 10.1.253.9.

Parte 2: Ethernet

5. Dada la siguiente trama Ethernet realice un análisis completo indicando los campos relevantes de la trama y del nivel de red involucrado.

```

00 20 78 e1 5a 80 00 a0 cc 30 c8 db 08 00 45 0b
00 40 f6 00 40 00 80 06 f0 a5 0a 02 00 02 0a 02
00 01 04 1e 00 15 00 4a 19 45 00 00 00 00 b0 02
20 00 e4 38 00 00 02 04 05 b4 01 03 03 00 01 01
08 0a 00 00 00 00 00 00 00 00 01 01 04 02 8f ec
d3 4f
  
```

6. Explique los motivos por los cuales se fijaron longitud máxima y mínima en 802.3.

Se fija una longitud mínima a efectos de asegurarse que todas las estaciones detecten colisión en caso de ocurrir esta. Con un paquete de longitud menor, podría darse el caso que, si dos estaciones distantes transmiten tramas y en una de ellas, la emisión del último bit de la trama se efectúa, antes de la recepción del primer bit de la trama de la estación colateral, la primera estación no detectaría la colisión, mientras que las estaciones intermedias si la detectan (pues recibieron bits de ambas tramas en simultáneo). La definición de una longitud mínima asegura que todas las estaciones detecten colisión y se opere en consecuencia.

La trama máxima tiene varios motivos: En particular porque un largo excesivo de trama implicaría la necesidad de poseer importante cantidad de almacenamiento temporal, sobre todo en el caso de ventanas de transmisión

grandes. En la actualidad eso no es un problema debido al bajo costo de la memoria, pero si lo era en los momentos de definición de la norma.

Por otro lado, la longitud máxima del paquete está acotada a efectos que una determinada estación no monopolice el canal indefinidamente. De esta forma se garantiza que cada tanto tiempo (aproximadamente 1.2 ms como máximo, en caso de una velocidad de 10 Mbps) exista una conmutación entre estaciones. El valor de 1518 bytes, - excluyendo el preámbulo y el SFD -, se ha determinado como compromiso (ni tan largo, ni tan corto), para garantizar una cierta performance en el diligenciamiento de paquetes a través de la red.

7. En Fast-Ethernet, la trama mínima, ¿tiene el mismo valor? Justifique. ¿Qué ocurre en Gigabit-Ethernet?

En Fast ethernet half-duplex, la trama mínima tiene la misma longitud que en Ethernet 10 Mbps (64 bytes toda la trama). Para ello se especifica una longitud máxima del segmento de LAN hasta el Hub, de 100 metros. Con esta máxima longitud del cable permitida y la trama mínima, se garantiza que todos los nodos estarán conscientes de la colisión.

En el caso de Gigabit – Ethernet, debido a que se sigue utilizando CSMA/CD y el mismo formato de trama que Ethernet 10 Mbps y 100 Mbps, se efectúa lo que se dio en llamar extensión de portadora. Con esta técnica, se extiende el valor de la trama mínima hasta un valor de 512 bytes en half-duplex, a efectos que el tiempo de transmisión sea mayor que el de propagación y las estaciones detecten colisiones.

En full-duplex, en velocidades de 100 Mbps y 1Gbps, al no existir colisiones el tamaño mínimo se conserva en 64 bytes para toda la trama.

8. ¿Por qué pueden subsistir en un mismo segmento Ethernet II y 802.3?

Después del MAC Header (de 12 bytes, con las MAC address de destino y origen), existe un campo de 2 bytes. Este mismo campo, Ethernet II lo interpreta como un indicador del tipo de protocolo (Type Field), mientras que la norma IEEE 802.3, lo interpreta como un campo de longitud del campo de datos (Length Field).

La subsistencia es posible, porque el largo máximo del campo de datos siempre es menor que 1536 bytes (0x600), mientras que una indicación de tipo de protocolo, se ha elegido con un valor mayor (0x800 para IP protocol, 0x806 para ARP protocol).

Ello significa, que el software de la capa de enlace, interpretará ese mismo campo de una u otra forma, en base al rango de valores que posea y por ello, identificará si se trata la norma Ethernet II o el standard 802.3.

9. El número de colisiones en Ethernet:

- a) Depende de la cantidad de estaciones.
- b) Es menor si se utiliza un Hub con 10BT

- a) El número de colisiones depende de la cantidad de estaciones. En Ethernet se emplea CSMA/CD con algoritmo 1-persistente. En este algoritmo, simplemente se garantiza que exista colisión de todas las estaciones que desean transmitir, inmediatamente después que el canal queda libre y se toman luego previsiones para el ordenamiento temporal de la transmisión para cada nodo, empleando un algoritmo de espera exponencial binaria (**backoff**) que cada estación realiza independientemente.

En esencia, el algoritmo otorga ranuras temporales, que son múltiplos del tiempo de transmisión de $k \cdot 512$ bits, donde k toma valores $0, 1, 2, 3, \dots, 2^{m-1}$. El exponente m se duplica en cada colisión, hasta un máximo de 10 y esto provoca que el universo de valores k que pueden ir adoptando aleatoriamente los nodos, también se duplica. Precisamente, al inicio el valor k es muy pequeño y si existe un número grande de

estaciones intentando transmitir, las colisiones serán más probables y el número de reintentos aumentará, provocándose que también aumente el número k de ranuras temporales, a efectos que los intentos de transmisión de los distintos nodos se vayan distribuyendo ordenadamente en el tiempo y no ocurra colisión. Al final el nodo que elija el inferior valor de k , tiene éxito en la transmisión.

El resto de los nodos, al constatar una nueva ocupación del canal, reinician su backoff desde 0, repitiéndose el proceso hasta que todos los nodos tengan éxito en su transmisión.

En definitiva, como puede verse la cantidad de colisiones depende directamente del número de estaciones y es necesario aumentar el número de turnos para transmitir. Obviamente, esa iteración en la búsqueda del valor de k necesario, provoca demora e ineficiencia en el canal cuando aumenta el número de estaciones en la red.

- b) Un Hub 10BT no reduce el número de colisiones debido a que todos los segmentos de la LAN que interconecta el Hub, quedan perteneciendo al mismo **dominio de colisión**, esto es, el Hub propaga la trama de bits de una estación al resto conectado, independientemente del hecho que alguna de las otras estaciones esté transmitiendo. Un Bridge en cambio, reduciría el número de colisiones, creando distintos dominios de colisión.