Departamento de
Automática y Computación
Automatika eta
Konputazio Saila

Campus de Arrosadía
Arrosadiko Campusa
31006 Pamplona - *Iruñea*
Tfno. 948 169113, Fax. 948 168924
Email: ayc@unavarra.es

Práctica 10 – Prestaciones en TCP

1- Objetivos

Análisis de TCP (Transmission Control Protocol) y prestaciones.

2- Material

Para la realización de esta práctica necesitaremos el siguiente equipamiento de los armarios:

- 3 PCs.
- 1 Concentrador Ethernet.
- 1 Conmutador Ethernet
- 4 cables categoría 5

3- Avisos generales

En los ordenadores dispuestos para la realización de estas prácticas (PC A, B y C) se ha creado una cuenta de nombre `arss` y password `telemat`. Esta cuenta tiene permisos para ejecutar mediante el comando `sudo` ciertos comandos restringidos normalmente al superusuario.

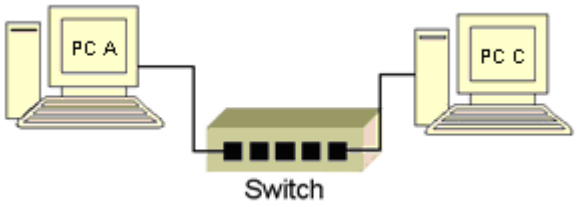
Si quieren conservar cualquier fichero entre sesiones guárdenlo en una memoria USB, dado que no se asegura que los ficheros creados o modificados durante una sesión de prácticas se mantengan para la siguiente.

4- Estados de una conexión TCP

Establecimiento

Configure en PCA y PCC una tarjeta de red, cada una con una dirección IP en el rango de direcciones 10.3.armario.0/24: `$ sudo ifconfig eth0 <ip> netmask <máscara>`

Conecte PCA y PCC mediante el switch0 de su armario (consulte la documentación sobre los armarios):



Una aplicación que funciona como servidor TCP, se programa para que pueda aceptar conexiones. Hasta que haya alguna conexión de algún cliente la aplicación servidor TCP se encontrará en estado LISTEN y cambiará de estado en el momento en el que el cliente se conecte.

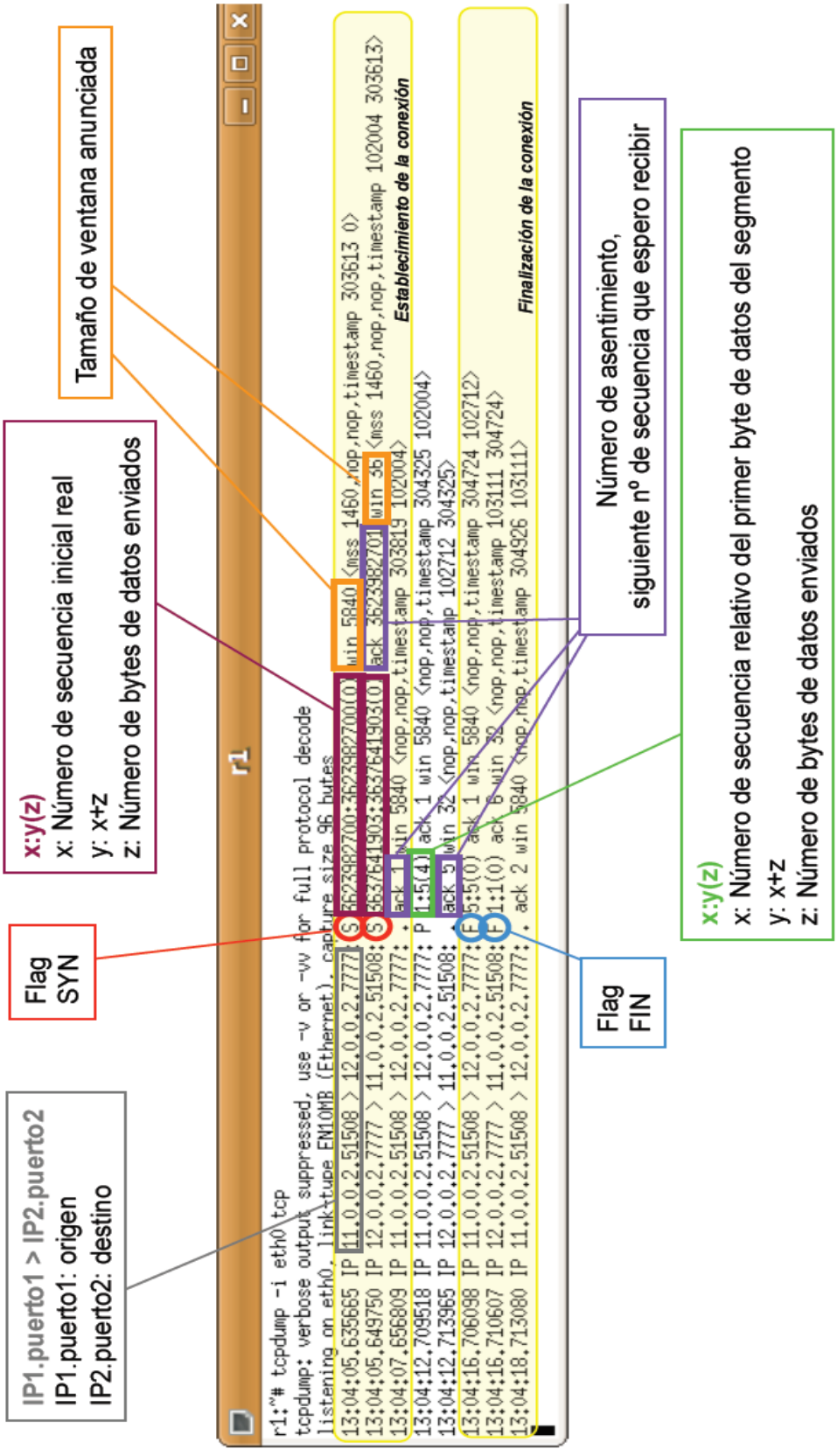
- Ejecute tcpdump en PC A, en su interfaz eth0, especificando que sólo capture el tráfico TCP:

```
$ tcpdump -i eth0 tcp
```

Departamento de
Automática y Computación
Automatika eta
Konputazio Saila

Campus de Arrosadía
Arrosadiko Campusa
31006 Pamplona - *Iruñea*
Tfno. 948 169113, Fax. 948 168924
Email: ayc@unavarra.es

Ensanche la ventana del terminal todo lo que pueda para ver cada segmento capturado en una sola línea. Para interpretar la salida de `tcpdump` utilice la información de la siguiente figura:



Campus de Arrosadía
Arrosadiko Campusa
31006 Pamplona - *Iruñea*
Tfno. 948 169113, Fax. 948 168924
Email: ayc@unavarra.es

- Arranque una aplicación servidor TCP que espere conexiones en el puerto 7777 en PC C, utilizando las herramientas `netem`, de la siguiente forma:

  ```
  $ nc -l -p 7777
  ```

- Para ver en qué estado se encuentra la conexión TCP en el lado del servidor (PC C) utilice el comando `netstat` de la siguiente forma:

  ```
  $ watch -n 0.5 netstat -nat
  ```

  El comando `watch` permite ejecutar repetidamente el comando que se escriba a continuación, en este caso `netstat`. La opción `-n` de `watch` permite especificar cada cuánto tiempo se repite la ejecución, en este caso, cada 0.5 segundos.

  Antes de continuar es importante que consulte el manual de `netstat` para entender bien el significado de cada una de las columnas que aparecen al ejecutarlo.

  El cambio de estado se produce muy rápidamente y no es posible apreciarlo, por ello es necesario introducir un cierto retardo en la comunicación PC A – PC C. Teniendo en cuenta que debe pasar de PC A a PC C (donde observará el cambio de estado), con un *retardo de diez segundos en la interfaz de PC A* será suficiente. Para ello realice el siguiente comando:

  ```
  $ sudo tc qdisc add dev eth0 root netem delay 10s
  ```

  Para saber algo más sobre `tc` consulte su manual (`man tc`)

- Coloque en la pantalla las ventanas de los terminales de `tcpdump`, `netstat` y nc, para que pueda visualizarlas simultáneamente.

- Observe en qué estado se encuentra el servidor antes de arrancar el cliente.

- Lance en PC A una aplicación cliente TCP que se conecte al servidor de PC C:

  ```
  $ nc <ipPC C> 7777
  ```

- Indique qué estados atraviesa el servidor hasta que el cliente se encuentra conectado.

- Observe en el tráfico capturado en PC C cómo se han intercambiado los 3 segmentos del establecimiento de la conexión.

- Apunte el valor de la ventana anunciada por el servidor.

Intercambio de datos

Desde PC A puede escribir tantos datos como quiera por la entrada estándar para enviar a PC C. Compruebe que los datos que envía PC A a PC C se quedan almacenados en el buffer de entrada de PC C hasta que la aplicación los lea:

- Si ha interrumpido la ejecución de tcpdump en PC C en su interfaz eth0, vuelva a lanzarlo.

- Detenga la ejecución de `nc` en el lado servidor con **`Ctrl+z`**, ésto provocará que la ejecución del servidor quede parada (pero no finalizada) y por tanto no realizará ninguna operación más, en particular, no leerá la información que reciba de TCP.

Campus de Arrosadía
Arrosadiko Campusa
31006 Pamplona - *Iruñea*
Tfno. 948 169113, Fax. 948 168924
Email: ayc@unavarra.es

- Si ha interrumpido la ejecución de netstat en el lado servidor (PC C), vuelva a lanzarlo mediante: `$ watch -n 0.5 netstat -nat`

- En el proceso `nc` del cliente en PC A introduzca una cadena de caracteres larga por la entrada estándar (por ejemplo pulse un rato la tecla de alguna letra hasta que aparezcan 2 líneas de esa letra por pantalla y después pulse la tecla `enter`). Esta cadena de caracteres se recibirá en la implementación de TCP en el lado servidor, pero la aplicación no leerá estos datos porque se encuentra detenida.

Checkpoint 10.1: Visualice que los datos se han quedado almacenados en el buffer de entrada (`Recv-Q`) del lado servidor. Justifique el valor de `Recv-Q` teniendo en cuenta que ha pulsado más caracteres para enviar en el cliente.

- Observe qué es lo que está ocurriendo con la captura de tráfico de PC C.

- Traiga a primer plano la ejecución de `nc` en PC C, mediante el comando `fg`, y la ejecución de la aplicación servidor continuará. Observe cómo la aplicación servidor ha leído los datos del buffer de entrada (ahora estará a cero) y los ha mostrado en la pantalla.

Finalización

Para ver los estados por los que atraviesa el servidor cuando el cliente cierra la conexión, sigue los siguientes pasos:

- Si ha interrumpido la ejecución de tcpdump en PC C en su interfaz eth0, vuelve a lanzarlo.

- Interrumpa la ejecución del cliente en PC A con `Ctrl+c`. La aplicación cliente mandará un segmento con el `flag de FIN` activado.

- El servidor `nc` está programado para terminar si el cliente le manda un segmento con el `flag FIN` activado, por este motivo, la aplicación nc en PC C finaliza la conexión mandando su segmento `FIN+ACK`. Cuando el servidor recibe el último `ACK` de PC A da por terminada la comunicación y finaliza su ejecución.

- En PC A aunque la aplicación ha terminado, los recursos de la conexión TCP tardan un tiempo en liberarse y si ejecuta en PC A: `$ watch -n 0.5 netstat -nat`

  Observará que la conexión TCP tarda un tiempo en liberarse. Fíjese en el estado en el que se encuentra. ¿Por qué se mantiene tanto tiempo en ese estado?

- En PC C la aplicación ha terminado y se han liberado inmediatamente los recursos de la conexión TCP. Si ejecuta en PC C (si lo había cerrado): `$ watch -n 0.5 netstat -nat`

  Observará que hay no hay conexiones.

Checkpoint 10.2: Justifique el estado TIME_WAIT en el que permanece PC A cuando el servidor PC C ya ha liberado su socket. ¿Cuánto tiempo permanece en este estado?

Departamento de
Automática y Computación
Automatika eta
Konputazio Saila

Campus de Arrosadía
Arrosadiko Campusa
31006 Pamplona - *Iruñea*
Tfno. 948 169113, Fax. 948 168924
Email: ayc@unavarra.es

5- Prestaciones TCP

- Utilice el mismo escenario del apartado anterior.
- Retorne a cero (0s) el retardo introducido en la interfaz eth0 de PC A.
- Cree un fichero, de un tamaño determinado, para transferir:

  ```
  $ dd if=/dev/zero of=unmega bs=1M count=1
  ```

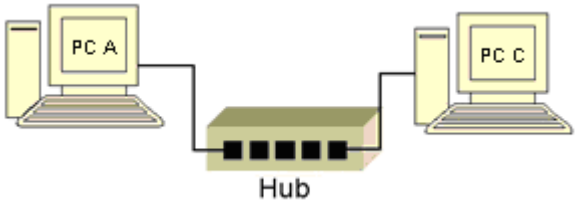
  "`unmega`" tendrá un tamaño de ¡1MB!. *(Importante: Borrarlo al finalizar la práctica)*

- Lance Wireshark en PC C y desmarque la opción `Edit, Preferences, Protocols, TCP, Relative Sequence Numbers $ Windows Scaling`.
- Realice una transferencia del fichero creado copiándolo de PC A a PC C mediante scp:

  ```
  $ scp unmega arss@<ip>:/home/arss
  ```

  Le solicitará la password del usuario `arss` en el otro equipo(PCs A, B y C "`telemat`").

- Detenga la captura de Wireshark, *guárdela* y analícela (*ordene las tramas por su marca de tiempo, pulsando en la pestaña* ***Time***), anotando los siguientes **valores**:
  - Valor de la ventana anunciada al comienzo de la conexión en cada uno de los sentidos.
  - MSS anunciado en las cabeceras opcionales en los dos sentidos. Justifica este valor teniendo en cuenta que el tamaño máximo del campo de datos de una trama Ethernet es de 1500 bytes.
  - RTT para un par de segmentos que envía el cliente al servidor. Debería coincidir con el obtenido mediante el comado `ping`. Verifíquelo.
  - Duración de la transferencia.
  - Estimación del valor medio, aproximado por inspección de la captura de Wireshark, de la ventana TCP en cada uno de los dos sentidos.
- A continuación realice el mismo experimento de transferencia del fichero "unmega" configurando, como en el apartado 4, un servidor en PC C en el puerto 7777 y un cliente en PC A mediante `nc`. En el caso del cliente: `$ nc <ipPC C> 7777 < unmega`
- *Guarde* la captura correspondiente de Wireshark en PC C y analícela anotando los mismos **valores**.
- A continuación repita este mismo experimento de transferencia del fichero "unmega" mediante los comandos `scp` y `nc` en el escenario siguiente:



- Analice los resultados obtenidos planteándose las siguienes preguntas:
    - ¿Por qué la diferencia de tiempos de transferencia entre scp y nc en un mismo escenario y un mismo archivo?
    - ¿Y entre escenarios?
    - ¿A que se deben las retransmisiones observadas en la captura de Wireshark en el segundo escenario?

Checkpoint 10.3: Seleccione una cualquiera de las retransmisiones presentes en sus capturas de tráfico e indique el segmento original a la que corresponde, especificando su número de secuencia y la confirmación del receptor una vez ha recibido dicha retransmisión con éxito.

Anexo

En Wireshark, además de mirar el contenido de los paquetes de una conexión TCP, puede verse en una gráfica la evolución del envío de datos y recepción de acks respecto al tiempo.

Wireshark permite mostrar varios tipos de gráficas de una conexión TCP.

Considere para el análisis de sus capturas la aplicación de las estadísticas de Wireshark, en concreto las correspondientes a:

- Round Trip Time Graph
- Throughput Graph
- Time-Sequence Graph (tcptrace)

Se detalla a continuación la gráfica de tcptrace.

Como una conexión TCP permite el envío de datos en ambos sentidos, se pueden visualizar 2 gráficas de tcptrace diferentes: las correspondientes a cada sentido de la comunicación.

Para ver en Wireshark la gráfica tcptrace de uno de los sentidos de una conexión TCP:

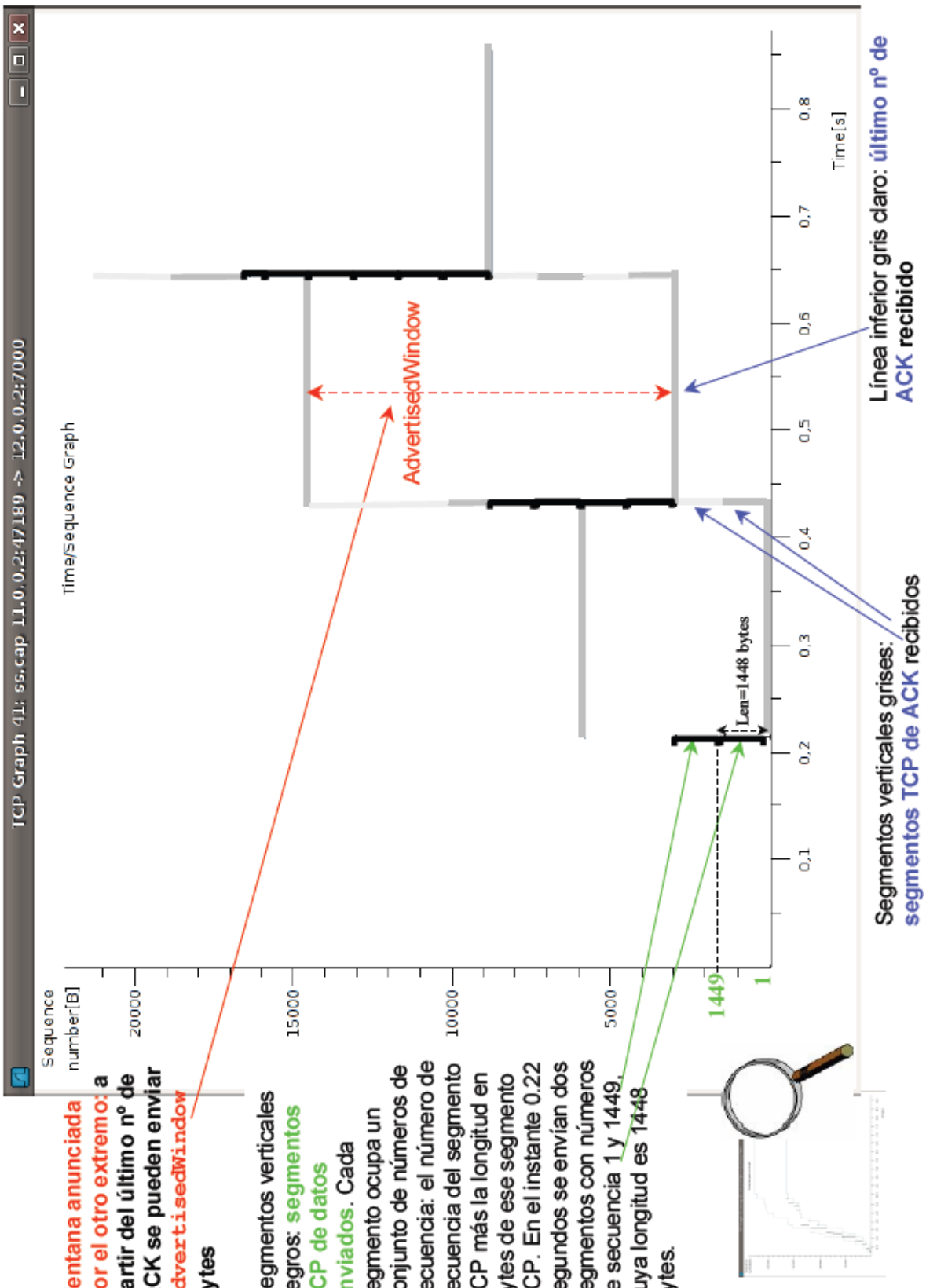
1. Carge el fichero de una captura que contenga los paquetes de una conexión TCP.

2. Seleccione un segmento de la conexión del sentido de la comunicación que quiere analizar (si el segmento seleccionado va del proceso A al proceso B, la gráfica que se mostrará será la correspondiente al envío de datos de A a B).

3. Seleccione en el menú de Wireshark: Statistics, TCP Stream Graph, Time-Sequence Graph (tcptrace)

Departamento de
Automática y Computación
Automatika eta
Konputazio Saila

Campus de Arrosadía
Arrosadiko Campusa
31006 Pamplona - *Iruñea*
Tfno. 948 169113, Fax. 948 168924
Email: ayc@unavarra.es



Departamento de
Automática y Computación
Automatika eta
Konputazio Saila

Campus de Arrosadía
Arrosadiko Campusa
31006 Pamplona - *Iruñea*
Tfno. 948 169113, Fax. 948 168924
Email: ayc@unavarra.es

Acciones sobre la gráfica *tcptrace*

Click central: zoom in

MAYS + Click central: zoom out

Arrastrar con el botón derecho: desplaza el gráfico (útil si se ha hecho \zoom in")

ESPACIO: activa/desactiva una cruz para ayudar a ver sobre los ejes la posición del ratón.

Click izquierdo sobre un segmento: selecciona el paquete concreto en la lista de paquetes de Wireshark.

CTRL + arrastrar con el botón derecho: lupa

s: Alterna entre números de secuencia relativos y absolutos, sólo si está desactivada la opción:

Edit, Preferences, Protocols, TCP, Relative sequence numbers and window scaling.