# Práctica 7

# Midiendo throughput en Ethernet

## 1. Objetivos

El objetivo de esta práctica es familiarizarse con los equipos de red de área local Ethernet y medir y comprender el throughput en una red real.

## 2. Throughput entre dos PCs [estimado 40 minutos]

En esta práctica usaremos de nuevo los puestos del Laboratorio de Telemática 1 con armarios rack y equipo de red.

En esta primera fase conectaremos un PC a otro directamente para comprobar el límite de velocidad de la tarjeta ethernet y posteriormente pondremos redes más complejas entre ellos para ver el efecto.

En primer lugar utilizaremos PC-A para enviar a PC-B. Identifica los puertos de red en el armario que corresponden con `eth0` de PC-A y PC-B (recuerda la práctica anterior con estos equipos) y únelos con un cable cruzado (en el laboratorio normalmente azules).

Configura la dirección IP `10.0.0.1` en PC-A y la `10.0.0.2` en PC-B (de nuevo explicado en una práctica anterior).

Comprueba que lo has hecho correctamente haciendo `ping` entre PC-A y PC-B.

En PC-A:

```
$ ping 10.0.0.2
```

En PC-B:

```
$ ping 10.0.0.1
```

Si no funciona asegúrate de arreglarlo antes de continuar.

Utilizaremos la herramienta `iperf` para enviar datos continuamente entre los PCs y poder observar el tráfico y medir el throughput. `iperf` funciona lanzándose en los ordenadores origen y destino de la información. Los programas se comunican a través de la red y envían la cantidad de información que indiquemos. El propio `iperf` nos da su propia medida del throughput que consigue enviar pero lo verificaremos observando la red con `wireshark`.

En el ordenador destino debemos lanzar `iperf` en modo servidor para que reciba los datos que le envíen desde el resto de `iperf`. `iperf` puede recibir datos utilizando un

protocolo de transporte fiable llamado TCP o bien utilizando un protocolo de transporte no fiable llamado UDP. El transporte fiable se asegura de que los datos llegan al otro lado y no se pierdan por el camino, retransmitiendo si fuese necesario. Al transporte no fiable le da igual que se pierdan, por lo que conseguirán diferentes resultados. Un ejemplo que nos encontramos hoy en día de TCP es cuando queremos recibir una página web ya que queremos asegurarnos que se transmite toda la información entre origen a destino; si faltase algún paquete no podríamos leer parte de esa página. Por otro lado se usa UDP en conexiones de vídeo en tiempo real ya que aunque se pierda parte de la información, podemos continuar reproduciendo el vídeo.

Los protocolos de transporte utilizan lo que se llaman valores de puerto para identificar los diferentes programas que están usando esos protocolos. Los puertos son simplemente números del 0 al 65535 ($2^{16}$-1).

Pon en PC-B un servidor iperf para recibir datos con transporte fiable en el puerto 10000. Eso se hace con el comando:

```
$ iperf -s -p 10000
```

Verás en la pantalla que el programa está esperando a que le lleguen datos. Esperará indefinidamente e imprimirá por pantalla estadísticas cada vez que le lleguen datos. De momento déjalo esperando en un terminal.

Pon en otro terminal de PC-B otro servidor `iperf` para recibir datos con transporte no fiable en el puerto 20000. Eso se hace con:

```
$ iperf -s -p 20000 -u
```

Déjalo esperando en tu terminal. Con esto ya tenemos programas que esperan y reciben los datos que les envíen desde otro ordenador.

Enviemos datos desde PC-A. Lanza `iperf` desde PC-A en modo cliente. En ese modo enviará datos continuamente a un servidor indicado. Continuamente, pero sólo durante el tiempo que le indique. Por ejemplo, para enviar datos con transporte fiable al servidor anterior lance en PC-A:

```
$ iperf -c 10.0.0.2 -p 10000 -t 10
```

Esto hará que durante 10 segundos el programa envíe datos continuamente hacia PC-B. Los datos se enviarán con un protocolo de ventana deslizante de los que se ven en teoría y se confirmarán para asegurarse de que llegan. Pero lo que nos importa es ver cuántos datos atraviesan la red. Observa que tanto el `iperf` cliente como el servidor nos dicen cuánto *throughput* en Mbps han conseguido enviar pero queremos ser capaces de medirlo de forma independiente.

Lanza `wireshark` en PC-A y en PC-B. Pon cada uno a capturar todo el tráfico de red que se vea en `eth0`. Lanza entonces el cliente durante 10 segundos y espera a que termine. Detén los wiresharks y observa el throughput en Mbps que han visto. ¿Cuánto es el throughput que ha observado? ¿Coincide con el que dice `iperf`? ¿Por qué ese throughput? ¿De qué velocidad dirías que es la tarjeta Ethernet de PC-A y PC-B?

Si usamos `iperf` con transporte fiable, `iperf` intenta enviar a toda la velocidad que puede y vemos los límites del protocolo de transporte fiable empleado. Si usamos `iperf` con transporte no fiable es posible que enviemos datos tan deprisa que se pierdan. `iperf` nos deja limitar esta velocidad controlando la carga que introduce el programa en la red. Usa ahora el transporte no fiable para enviar datos al servidor. El comando es parecido, sólo que con la opción `-u` indica que es transporte UDP y puedes usar la opción `-b` para especificar la velocidad de envío. Enviemos a varias velocidades para probar:

```
$ iperf -c 10.0.0.2 -p 20000 -u -t 10 -b 4Mbps

$ iperf -c 10.0.0.2 -p 20000 -u -t 10 -b 40Mbps

$ iperf -c 10.0.0.2 -p 20000 -u -t 10 -b 150Mbps
```

Observa los resultados obtenidos. Realiza un experimento para medir esto y presentarlo al profesor. Prepara los wiresharks en el PC-A y PC-B y captura una traza en la que se vean en sucesión rápida las tres pruebas de antes con 4Mbps, 40Mbps y 150Mbps. Cuando tenga las trazas en el origen y el destino, dibuja la gráfica del throughput en la que se vean los tres envíos y se pueda ver los throughputs obtenidos. Cuando tengas esta gráfica abre un terminal y haz:

```
$ sudo ifconfig eth0
```

Coloca el terminal de forma que se vea el resultado y la gráfica correctamente. Haz una captura de pantalla. Graba la captura de pantalla de PC-A y PC-B con los nombres **cp1-origen.png** y **cp1-destino.png**. Estos son los ficheros que deberás subir para probar que has realizado el checkpoint 1. Si quieres puedes mostrar primero el checkpoint al profesor de prácticas y generar los archivos cuando le haya dado el visto bueno.

| Checkpoint 1. Muestra tus resultados al profesor y sube los ficheros |
|---|

Si ha llegado hasta aquí llama al profesor de prácticas y muéstrale los throughputs que has obtenido. Explícale si tienen sentido y contesta a las preguntas.

# 3. Throughput atravesando Ethernet [estimado 40 minutos]

Prepararemos un escenario en el que haya elementos de red entre PC-A y PC-B como se ve en la figura 1. Queremos que PC-A y PC-B estén en redes Ethernet de 100Mbps pero que estas dos redes estén unidas entre si por un enlace a 10Mbps. Para eso se han configurado unos puertos a 10Mbps en los conmutadores del laboratorio. **Debes usar exactamente los conmutadores y puertos indicados**.

En primer lugar prueba los dos PCs en el mismo conmutador. Conecta el puerto `eth0` de PC-A al conmutador Ethernet (marca D-Link) inferior del armario. Conecta `eth0` de PC-B a otro puerto del mismo switch (que no sean los puertos últimos 23 y 24). Vuelve a realizar los experimentos anteriores con `iperf` y compruebe que dan el mismo resultado. Si ha elegido puertos del switch a 100Mbps no debería haber diferencia.

Configura ahora el escenario de la figura 1. PC-A ya está en el switch correcto. Cambia PC-B al switch de la práctica anterior (el verde de marca OVISLINK). Usa uno de los puertos del 1 al 7. Ahora no debería poder hacer `ping` de PC-A a PC-B porque falta unir los conmutadores. Une el puerto 8 del conmutador superior (el verde de marca OVISLINK) con el puerto 23 ó 24 del conmutador inferior (marca D-Link). Al hacer esto deberías poder hacer `ping`. Asegúrate de que esto funciona correctamente antes de seguir.

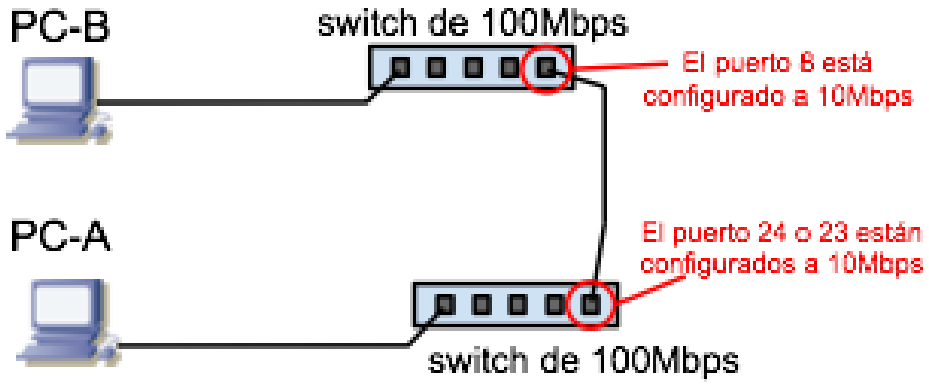


*Figura 1*

Una vez acabada la configuración, realiza el experimento anterior nuevamente con transporte no fiable y velocidades de 3Mbps, 8Mbps y 15Mbps. Obtén las gráficas en PC-A y PC-B. ¿Por qué ahora no son iguales?

Genera las pruebas de haber realizado este experimento (capturas de pantalla) como en el caso anterior. Guárdalas con el nombre **cp2-origen.png** y **cp2-destino.png**. Estos son los ficheros que deberás subir para probar que ha realizado el checkpoint 2.

| Checkpoint 2. Subir los ficheros del segundo experimento. No es necesario mostrárselo al profesor de prácticas |
|---|

# 4. Escenario avanzado con más hosts [estimado 40 minutos]

Conecta el PC-C al mismo switch que PC-A (usa cualquier puerto excepto el 23 y 24). Configura el PC-C con la dirección 10.0.0.3. Lanza en PC-C tres envíos de transporte no fiable con `iperf` hacia el servidor de PC-B. Cada envío debe mantener 2Mbps indefinidamente. Esto último se puede hacer, a efectos prácticos, poniéndoles por ejemplo 1h de duración.

Prepara capturas con `wireshark` en los tres ordenadores para lanzarlas cuando tengas listo lo siguiente.

En PC-A prepara un envío con transporte fiable hacia PC-B que dure unos 20 segundos. Lanza las capturas. Espera 10 segundos y lanza el envío. Después de que acabe el envío, espere otros 10 segundos y para todas las capturas.

Haz las gráficas del throughput y compáralas. Interpreta si tienen sentido los throughputs obtenidos. Guarda las capturas de PC-A y PC-B como **cp3-origen.png** y **cp3-destino.png**. Estos son los ficheros que deberás subir para probar que ha realizado el checkpoint 3.

| Checkpoint 3. Subir los ficheros del tercer experimento |
|---|

Muestra al profesor los resultados del checkpoint 2 y 3.