

**Primera Práctica de Laboratorio. Entrega Obligatoria: 27 de noviembre de 2020****BLOQUE N° 1. Programación con sockets****Desarrollo de un servicio estándar: TFTP**

Protocolo de transferencia de ficheros trivial (TFTP). Implementar un proceso servidor y un proceso cliente que realicen la transferencia de ficheros sobre UDP de acuerdo al protocolo estándar TFTP definido en el RFC 1350. Tener en cuenta los siguientes puntos.

- Usar otro puerto diferente al estándar 69 de TFTP para el servidor.
- La transferencia podrá hacerse en cualquiera de los dos sentidos (*get* o *put*). La fase de transferencia de datos y su finalización es igual para ambos procesos cliente y servidor, variando sólo el establecimiento de la comunicación.
- El fichero que se transfiera debe ser textual ya que se mostrará por pantalla (opcionalmente pueden almacenarse los datos transferidos en un fichero).
- El proceso cliente tendrá un pequeño interfaz de usuario para pedir el tipo de la transferencia y el nombre del fichero a transferir. Emular los comandos del cliente tftp estándar (como mínimo: *connect*, *get*, *put* y *quit*).
- Tanto el cliente como el servidor comprobarán que todos los mensajes que le llegan han sido enviados por el proceso correcto (comprobando el identificador de transferencia que será la IP y el puerto), sino contestará con un mensaje de error. El servidor podrá ser concurrente o iterativo.
- Ambos procesos implementarán los mecanismos para garantizar que la transferencia se haga de forma fiable. Para ello se comprobará que los números de secuencia de los paquetes sean consecutivos y ante la pérdida de un paquete se retransmitirá después de esperar 1 segundo. Poner un número máximo de reintentos (ej: 3 o 5).
- Cada proceso tendrá dos modos funcionamiento: el correcto y con pérdida de paquetes. El modo de funcionamiento con pérdida de paquetes consiste en que durante el inicio y la transferencia de un fichero se podrán perder paquetes (acks o datos) hasta un 5%. La pérdida de paquetes se hará simulada, cada vez que un proceso cliente o servidor tenga que enviar un mensaje, de forma aleatoria decidirá si lo envía o no. Al finalizar se dará un resumen estadístico con el numero/porcentaje de paquetes perdidos y retransmitidos.
- Por simplicidad no se considerará que el primer paquete enviado (READ o WRITE) pueda perderse.
- El proceso cliente (y/o el servidor) dará la traza de la transferencia con el formato que se sigue en el ejemplo:

```
----> RRQ "fichero.dat" "octet"
----> DATA 1 512 bytes
----> DATA 2 10 bytes
----> DATA 2 10 bytes (R)
```

```
<---- ACK 0
<---- ACK 1
<---- ACK 2 (P)
<---- ACK 2
```