# Protocolos De Internet

## Trabajo Práctico 1 - Parte C - 1)

El archivo ejecutable *select* es un servidor TCP con concurrencia aparente. Una vez que lo ejecutamos, el servidor va a estar a la espera de conexiones entrantes. Como cliente usaremos Telnet, que no es ni mas ni menos que un cliente genérico de TCP (no hace falta desarollar el cliente para poder realizar este prueba). Podemos recibir o mandar mensajes.

Comenzamos con la descripcion de la experiencia. Ejecutamos *select* por consola y nos apareció lo siguiente:

```
server:$ ./select
Server-socket() OK...
Server-setsockopt() OK...
Server-bind() is OK...
Server-listen() OK...
```

El servidor ya está listo y "espera" (en realidad espera a que uno de los sockets pasados por lista tengan algo para leer, luego el accept se realiza sin bloqueo) a solicitudes de conexión. Esta escuchando en el puerto 2020. Activamos el cliente y procedemos a observar con el Wireshark (filtramos con tcp.port == 2020). Ejecutamos *telnet localhost 2020* en otra consola y nos apareció lo siguiente:

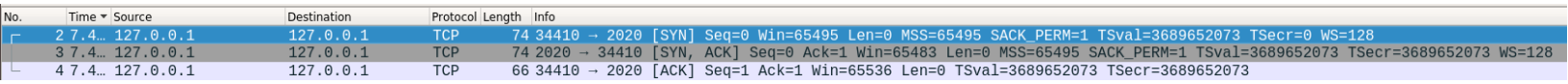
```
telnet localhost 2020
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
```

En la consola del servidor veremos:

```
Server-socket() OK...
Server-setsockopt() OK...
Server-bind() is OK...
Server-listen() OK...
Server-select() K...
Server-accept() OK...
./select: Nueva conexión desde 127.0.0.1 en socket 4
Server-select() K...
```

Significa que ya hemos establecido una conexión con el servidor y el cliente ya puede comunicarse con este mismo.

Esto se ve en el wireshark:



| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 2 | 7.4… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 34410 → 2020 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=3689652073 TSecr=0 WS=128 |
| 3 | 7.4… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 2020 → 34410 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=3689652073 TSecr=3689652073 WS=128 |
| 4 | 7.4… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 34410 → 2020 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3689652073 TSecr=3689652073 |

Como es un servidor que emplea concurrencia aparente, la función `select()` puso al socket descriptor asociado al socket que esta recibiendo nuevas conexiones en la lista de los file descriptors que estan listos para entregar información. Es decir, está listo para realizar una operacion como `read()` sin interrupcion. El servidor reconoce que el Socket recibió una conexión y ejecuta el `accept()`, se crea un nuevo socket y esatablecemos la conexión. El primer cliente que se conectó utilizó el puerto 34410 para conectarse a la 2020 del servidor. Vamos a agregar 4 clientes mas (es decir, cuatro consolas mas).

Procedemos a ver al servidor y obtendremos lo siguiente:

```
server:$ ./select
Server-socket() OK...
Server-setsockopt() OK...
Server-bind() is OK...
Server-listen() OK...
Server-select() K...
Server-accept() OK...
./select: Nueva conexión desde 127.0.0.1 en socket 4
Server-select() K...
Server-accept() OK...
./select: Nueva conexión desde 127.0.0.1 en socket 5
Server-select() K...
Server-accept() OK...
./select: Nueva conexión desde 127.0.0.1 en socket 6
Server-select() K...
Server-accept() OK...
./select: Nueva conexión desde 127.0.0.1 en socket 7
Server-select() K...
Server-accept() OK...
./select: Nueva conexión desde 127.0.0.1 en socket 8
```

¿Por que arranca desde 4 y no desde 1? La razón es que los primeros file descriptors son para los archivos estandar como lo es `stdin`,`stdout`,y `stderr`. El tercero es ocupado por el socket que esta recibiendo conexiones.

Observemos lo sucedido en el Wireshark:

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 2 | 7.4… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 34410 → 2020 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=3689652073 TSecr=0 WS=128 |
| 3 | 7.4… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 2020 → 34410 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=3689652073 TSecr=3689652073 WS=128 |
| 4 | 7.4… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 34410 → 2020 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3689652073 TSecr=3689652073 |
| 4161 | 285… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 35460 → 2020 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=3692501285 TSecr=0 WS=128 |
| 4162 | 285… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 2020 → 35460 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=3692501285 TSecr=3692501285 WS=128 |
| 4163 | 285… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35460 → 2020 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3692501285 TSecr=3692501285 |
| 4169 | 286… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 35464 → 2020 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=3692510503 TSecr=0 WS=128 |
| 4170 | 286… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 2020 → 35464 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=3692510503 TSecr=3692510503 WS=128 |
| 4171 | 286… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35464 → 2020 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3692510503 TSecr=3692510503 |
| 4172 | 286… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 35466 → 2020 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=3692513188 TSecr=0 WS=128 |
| 4173 | 286… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 2020 → 35466 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=3692513188 TSecr=3692513188 WS=128 |
| 4174 | 286… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35466 → 2020 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3692513188 TSecr=3692513188 |
| 4175 | 287… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 35468 → 2020 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=3692514759 TSecr=0 WS=128 |
| 4176 | 287… | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 2020 → 35468 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=3692514759 TSecr=3692514759 WS=128 |
| 4177 | 287… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35468 → 2020 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3692514759 TSecr=3692514759 |

Establecimos 4 conexiones mas que vienen de los puertos 25460, 35464, 35466, y 35468.

Vamos a probar ahora mandarle 'Hello World!' al servidor en uno de los clientes. Escribimos 'Hello World!' en uno de los clientes y la consola se va a ver de la siguiente manera:

```
telnet localhost 2020
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Hello World!
```

Si revisamos las consolas de los clientes vamos a ver que en todos se lee 'Hello World!' (se va a ver identico al pedazo de consola mostrado anteriormente). Revisamos el Wireshark y vemos los nuevos paquetes capturados:

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 5086 | 434… | 127.0.0.1 | 127.0.0.1 | TCP | 80 | 35464 → 2020 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=14 TSval=3693987644 TSecr=3692510503 |
| 5087 | 434… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 2020 → 35464 [ACK] Seq=1 Ack=15 Win=65536 Len=0 TSval=3693987644 TSecr=3693987644 |
| 5088 | 434… | 127.0.0.1 | 127.0.0.1 | TCP | 80 | 2020 → 34410 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=14 TSval=3693987644 TSecr=3689652073 |
| 5089 | 434… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 34410 → 2020 [ACK] Seq=1 Ack=15 Win=65536 Len=0 TSval=3693987644 TSecr=3693987644 |
| 5090 | 434… | 127.0.0.1 | 127.0.0.1 | TCP | 80 | 2020 → 35460 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=14 TSval=3693987644 TSecr=3692501285 |
| 5091 | 434… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35460 → 2020 [ACK] Seq=1 Ack=15 Win=65536 Len=0 TSval=3693987644 TSecr=3693987644 |
| 5092 | 434… | 127.0.0.1 | 127.0.0.1 | TCP | 80 | 2020 → 35466 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=14 TSval=3693987644 TSecr=3692513188 |
| 5093 | 434… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35466 → 2020 [ACK] Seq=1 Ack=15 Win=65536 Len=0 TSval=3693987644 TSecr=3693987644 |
| 5094 | 434… | 127.0.0.1 | 127.0.0.1 | TCP | 80 | 2020 → 35468 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=14 TSval=3693987644 TSecr=3692514759 |
| 5095 | 434… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35468 → 2020 [ACK] Seq=1 Ack=15 Win=65536 Len=0 TSval=3693987644 TSecr=3693987644 |

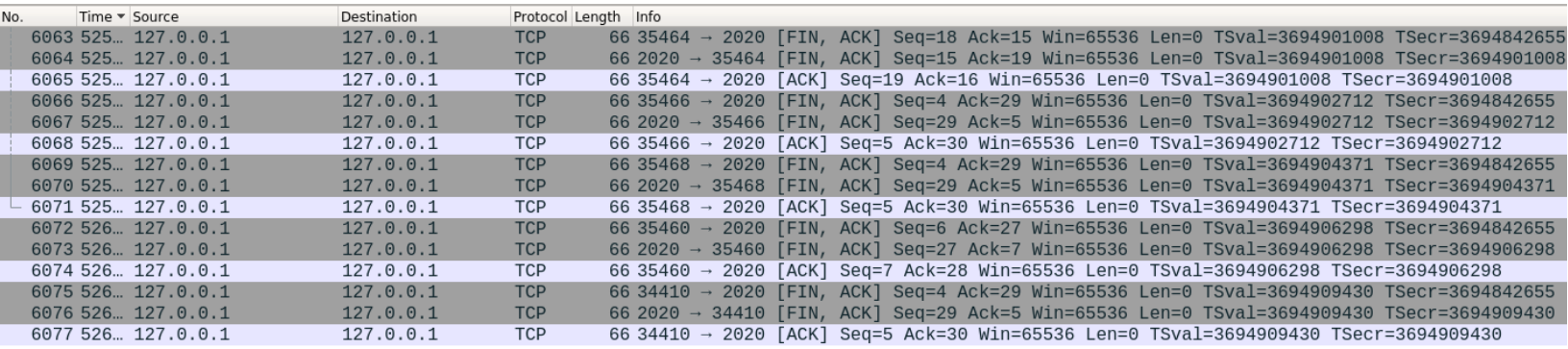
Si observamos el paquete No. 5086, el cliente le manda al servidor el mensaje, e inmediatamente despues el servidor manda ese mensaje a todos los otros clientes (esto podemos observarlo mirando despues del paquete No. 5088). Es por esa razon por la que observamos en todas las consolas el mensaje. `select()`puso al socket del cliente dentro de la lista de los sockets que tienen datos para leer (listos para leer sin interrupciones) y luego reenvia este mensaje a todos los clientes.

Procedemos a realizar la desconexión desde los clientes:

```
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Hello World!
^]
telnet>
```

Salimos del Telnet y los clientes mandaron el paquete con el FIN. Observamos que en la consola del servidor se agregaron estas lineas:

```
6./select: socket 6 desconectado
Server-select() K...
7./select: socket 7 desconectado
Server-select() K...
8./select: socket 8 desconectado
Server-select() K...
5./select: socket 5 desconectado
Server-select() K...
4./select: socket 4 desconectado
```



| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 6063 | 525… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35464 → 2020 [FIN, ACK] Seq=18 Ack=15 Win=65536 Len=0 TSval=3694901008 TSecr=3694842655 |
| 6064 | 525… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 2020 → 35464 [FIN, ACK] Seq=15 Ack=19 Win=65536 Len=0 TSval=3694901008 TSecr=3694901008 |
| 6065 | 525… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35464 → 2020 [ACK] Seq=19 Ack=16 Win=65536 Len=0 TSval=3694901008 TSecr=3694901008 |
| 6066 | 525… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35466 → 2020 [FIN, ACK] Seq=4 Ack=29 Win=65536 Len=0 TSval=3694902712 TSecr=3694842655 |
| 6067 | 525… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 2020 → 35466 [FIN, ACK] Seq=29 Ack=5 Win=65536 Len=0 TSval=3694902712 TSecr=3694902712 |
| 6068 | 525… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35466 → 2020 [ACK] Seq=5 Ack=30 Win=65536 Len=0 TSval=3694902712 TSecr=3694902712 |
| 6069 | 525… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35468 → 2020 [FIN, ACK] Seq=4 Ack=29 Win=65536 Len=0 TSval=3694904371 TSecr=3694842655 |
| 6070 | 525… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 2020 → 35468 [FIN, ACK] Seq=29 Ack=5 Win=65536 Len=0 TSval=3694904371 TSecr=3694904371 |
| 6071 | 525… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35468 → 2020 [ACK] Seq=5 Ack=30 Win=65536 Len=0 TSval=3694904371 TSecr=3694904371 |
| 6072 | 526… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35460 → 2020 [FIN, ACK] Seq=6 Ack=27 Win=65536 Len=0 TSval=3694906298 TSecr=3694842655 |
| 6073 | 526… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 2020 → 35460 [FIN, ACK] Seq=27 Ack=7 Win=65536 Len=0 TSval=3694906298 TSecr=3694906298 |
| 6074 | 526… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 35460 → 2020 [ACK] Seq=7 Ack=28 Win=65536 Len=0 TSval=3694906298 TSecr=3694906298 |
| 6075 | 526… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 34410 → 2020 [FIN, ACK] Seq=4 Ack=29 Win=65536 Len=0 TSval=3694909430 TSecr=3694842655 |
| 6076 | 526… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 2020 → 34410 [FIN, ACK] Seq=29 Ack=5 Win=65536 Len=0 TSval=3694909430 TSecr=3694909430 |
| 6077 | 526… | 127.0.0.1 | 127.0.0.1 | TCP | 66 | 34410 → 2020 [ACK] Seq=5 Ack=30 Win=65536 Len=0 TSval=3694909430 TSecr=3694909430 |

`select()` fue poniendo a los clientes dentro de las lista de los sockets que tienen mensajes, el servidor detecta que son para finalizar ela conexión (el entero recibido por `recv()` es menor o igual a 0) y procede a cerrarlo.

Todo lo realizado por el servidor fue en un solo proceso y la experiencia esta terminada.