

NOMBRE Y APELLIDOS DEL ALUMNO:

## Arquitectura de Redes

Laboratorio

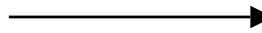
Prueba Final

- 1.- (0.75 puntos) Indicar la secuencia de comandos que permite descargar, por medio del protocolo FTP sobre la utilidad netcat, el archivo: /public/Docs/listado.txt en modo pasivo del servidor ftp.servidor.com. Si es preciso utilizar varios terminales, representar los comandos introducidos en cada uno y su precedencia de la siguiente manera.

### Terminal 1

> (Comando 1)

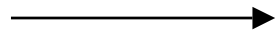
(Respuesta 1)



### Terminal 2

> (Comando 1)

(Respuesta 1)



### Terminal 3

> (Comando 1)

(Respuesta 1)

Lista de comandos del protocolo FTP que pueden ser de utilidad.

CWD

PASV

EPRT |1|a1,a2,a3,a4|puerto|

PORT a1,a2,a3,a4,p1,p2

EPSV

RETR

LIST

STAT

PASS

USER



- 2.- (0.75 puntos) El código siguiente es similar al del ejercicio 4 de la práctica 2. Se trata de un sencillo servidor que devuelve la suma de dos datos que el cliente envía. Se incluye el código del cliente y el del servidor. Modificar lo necesario (indicando qué y dónde) para que, en lugar de la suma, el servidor devuelva un 1 si es mayor el primero de los dos, un -1 si es mayor el segundo, y un 0 si son iguales. El cliente los mostrará por pantalla de mayor a menor según le indique el servidor.

```
/* Servidor.c */
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <string.h>
#include <unistd.h>

int main(int argc, char *argv[])
{
    struct sockaddr_in server_addr, client_addr;
    int c,i,sd, sc, val;
    unsigned int size;
    long int num[2], res;

    sd = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
    val = 1;
    setsockopt(sd, SOL_SOCKET, SO_REUSEADDR, (char *) &val, sizeof(int));

    bzero((char *)&server_addr, sizeof(server_addr));
    server_addr.sin_family = AF_INET;
    server_addr.sin_addr.s_addr = INADDR_ANY;
    server_addr.sin_port = htons(56789);

    bind(sd, (struct sockaddr *)&server_addr, sizeof(server_addr));
    listen(sd, 5);
    size = sizeof(client_addr);

    while (1)
    {
        printf("Esperando conexión\n");
        sc = accept(sd, (struct sockaddr *)&client_addr,&size);

        for(i=0;
            i<2*sizeof(long int)&&(c=read(sc, ((char *)num)+i,2*sizeof(long int)-i))>0;
            i+=c); /* recibe la petición */

        res = htonl(ntohl(num[0]) + ntohl(num[1]));
        write(sc, &res, sizeof(long int)); /* se envía el resultado */
        close(sc);
    }
    close (sd);
    return 0;
}

/* Cliente.c */
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <string.h>
#include <unistd.h>

int main(int argc, char* argv[])
{

```

```

int i,c,sd;
struct sockaddr_in server_addr;
struct hostent *hp;
long int num[2], res;

if (argc!=3)
{
    fprintf(stderr, "Uso: %s primer_sumando segundo_sumando\n", argv[0]);
    return 1;
}
sd = socket(PAF_INET, SOCK_STREAM, IPPROTO_TCP);

bzero((char *)&server_addr, sizeof(server_addr));
hp = gethostbyname ("localhost");

memcpy (&(server_addr.sin_addr), hp->h_addr, hp->h_length);
server_addr.sin_family = AF_INET;
server_addr.sin_port = htons(56789);
/* se establece la conexión */
connect(sd, (struct sockaddr *) &server_addr, sizeof(server_addr));

num[0]=htonl(atoi(argv[1]));
num[1]=htonl(atoi(argv[2]));

write(sd, (char *) num, 2 *sizeof(long int));    /*envía la petición*/

for(i=0;
    i<sizeof(long int)&&(c=read(sd,((char *)&res)+i,sizeof(long int)-i))>0;i+=c);
/* recibe la respuesta */

printf("El resultado es: %d \n", ntohl(res));
close (sd);
return 0;
}

```

Las siguientes 4 cuestiones se plantean en el contexto de la sesión FTP que se lleva a cabo, parte de cuya captura se muestra en la imagen. Responder a las cuestiones con Verdadero o Falso. En la línea 40 de la captura, lo que está ocurriendo es:

No.	Time	Source	Destination	Protocol	Length	Info
33	27.041977	192.168.1.36	193.146.58.130	FTP	72	Request: LIST
34	27.094804	193.146.58.130	192.168.1.36	TCP	74	ftp-data > 43425 [SYN] Seq=0 Win=5840 Len=0 MSS=1452 SACK_PERM=1 TSval=491183494 TSecr=
35	27.094860	192.168.1.36	193.146.58.130	TCP	74	43425 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PERM=1 TSval=
36	27.134574	193.146.58.130	192.168.1.36	TCP	66	ftp > 58430 [ACK] Seq=213 Ack=99 Win=5888 Len=0 TSval=491183534 TSecr=11065626
37	27.144632	193.146.58.130	192.168.1.36	TCP	66	ftp-data > 43425 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=491183544 TSecr=11065639
38	27.145860	193.146.58.130	192.168.1.36	FTP-DATA	218	FTP Data: 152 bytes
39	27.145931	192.168.1.36	193.146.58.130	TCP	66	43425 > ftp-data [ACK] Seq=1 Ack=153 Win=6912 Len=0 TSval=11065651 TSecr=491183544
40	27.145978	193.146.58.130	192.168.1.36	TCP	66	ftp-data > 43425 [FIN, ACK] Seq=153 Ack=1 Win=5888 Len=0 TSval=491183544 TSecr=11065652
41	27.146577	193.146.58.130	192.168.1.36	FTP	105	Response: 150 Here comes the directory listing.
42	27.146871	192.168.1.36	193.146.58.130	TCP	66	43425 > ftp-data [FIN, ACK] Seq=1 Ack=154 Win=6912 Len=0 TSval=11065652 TSecr=491183544
43	27.185936	192.168.1.36	193.146.58.130	TCP	66	58430 > ftp [ACK] Seq=99 Ack=252 Win=5888 Len=0 TSval=11065662 TSecr=491183544
44	27.196649	193.146.58.130	192.168.1.36	TCP	66	ftp-data > 43425 [ACK] Seq=154 Ack=2 Win=5888 Len=0 TSval=491183596 TSecr=11065652
45	27.197855	193.146.58.130	192.168.1.36	FTP	90	Response: 226 Directory send OK.

```

Frame 40: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
Ethernet II, Src: Comtrend_7d:aa:32 (00:1d:20:7d:aa:32), Dst: IntelCor_lc:ef:20 (00:21:5d:1c:ef:20)
Internet Protocol Version 4, Src: 193.146.58.130 (193.146.58.130), Dst: 192.168.1.36 (192.168.1.36)
Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 43425 (43425), Seq: 153, Ack: 1, Len: 0
  Source port: ftp-data (20)
  Destination port: 43425 (43425)
  [Stream index: 3]
  Sequence number: 153 (relative sequence number)
  Acknowledgement number: 1 (relative ack number)
  Header length: 32 bytes
  Flags: 0x11 (FIN, ACK)
    000. .... = Reserved: Not set
    ...0 .... = Nonce: Not set
    .... 0... = Congestion Window Reduced (CWR): Not set
    .... .0.. = ECN-Echo: Not set
    .... ..0. = Urgent: Not set
    .... ...1 .... = Acknowledgement: Set
    .... .... 0... = Push: Not set
    .... ..0.. = Reset: Not set
    .... .... .0. = Syn: Not set
    ... ..1 = Fin: Set
  Window size value: 46
  [Calculated window size: 5888]
  [Window size scaling factor: 128]
  Checksum: 0x5b03 [validation disabled]
  Options: (12 bytes)
  
```

TCP Options (tcp.options), 1... Packets: 6939 Displayed: 6939 Marked: 0 Load time: 0:00.749 Profile: Default

- (0.1 puntos)** El cliente está aceptando un cierre de conexión por parte del servidor.
  - (0.1 puntos)** El servidor está aceptando un cierre de conexión por parte del cliente.
  - (0.1 puntos)** El cliente está indicando un cierre de conexión al servidor.
  - (0.1 puntos)** El servidor está indicando un cierre de conexión al cliente.
- 
- (0.1 puntos)** En la captura anterior, ¿a qué línea está respondiendo la línea 41?