



Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
Redes de Computadores (CI-4835)

TALLER # 2

OBJETIVOS ESPECÍFICOS:

Al finalizar esta sesión experimental el estudiante deberá estar en capacidad de:

- Comprender el uso de la Interfaz de Aplicaciones (API) Sockets de Berkeley.

RECURSOS REQUERIDOS:

- Para la ejecución de esta práctica el estudiante deberá disponer un PC conectado a una red de computadores, ejecutando alguna distribución de GNU Linux y con la API “*Sockets*” instalada.

TIEMPO ESTIMADO: 2 horas

ACTIVIDADES A REALIZAR:

A.- Discuta el siguiente código de un servidor de Chat.

```
/*
 * Ejemplo de server de chat simple con datagramas (UDP).
 *
 * Leandro Lucarella - Copyleft 2004
 * Basado en otros ejemplos públicos.
 */

#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
#include <unistd.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <sys/types.h>

#define SERVER_PORT 4321
#define BUFFER_LEN 1024

int main(int argc, char *argv[])
{
    int sockfd; /* descriptor para el socket */
    struct sockaddr_in my_addr; /* direccion IP y numero de puerto local */
```

```

struct sockaddr_in their_addr; /* direccion IP y numero de puerto del cliente */

/* addr_len contendra el tamaño de la estructura sockaddr_in y numbytes el
 * numero de bytes recibidos */
int addr_len, numbytes;
char buf[BUFFER_LEN]; /* Buffer de recepción */

/* se crea el socket */
if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) == -1) {
    perror("socket");
    exit(1);
}

/* Se establece la estructura my_addr para luego llamar a bind() */
my_addr.sin_family = AF_INET; /* usa host byte order */
my_addr.sin_port = htons(SERVER_PORT); /* usa network byte order */
my_addr.sin_addr.s_addr = INADDR_ANY; /* escuchamos en todas las IPs */
bzero(&(my_addr.sin_zero), 8); /* rellena con ceros el resto de la estructura */

/* Se le da un nombre al socket (se lo asocia al puerto e IPs) */
printf("Asignado direccion al socket ....\n");
if (bind(sockfd, (struct sockaddr *)&my_addr, sizeof(struct sockaddr)) == -1) {
    perror("bind");
    exit(2);
}

/* Se reciben los datos (directamente, UDP no necesita conexión) */
addr_len = sizeof(struct sockaddr);
printf("Esperando datos ....\n");
if ((numbytes=recvfrom(sockfd, buf, BUFFER_LEN, 0, (struct sockaddr *)&their_addr,
(socklen_t *)&addr_len)) == -1) {
    perror("recvfrom");
    exit(3);
}

/* Se visualiza lo recibido */
printf("paquete proveniente de : %s\n", inet_ntoa(their_addr.sin_addr));
printf("longitud del paquete en bytes: %d\n", numbytes);
buf[numbytes] = '\0';
printf("el paquete contiene: %s\n", buf);

/* cerramos descriptor del socket */
close(sockfd);

exit (0);
}

```

2.- Discuta el siguiente código de un cliente de Chat.

```

/*
 * Ejemplo de cliente de chat simple con datagramas (UDP).
 *
 * Leandro Lucarella - Copyleft 2004
 * Basado en diversos ejemplos públicos.
 *
 */

```

```

#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
#include <unistd.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <sys/types.h>

#define SERVER_PORT 4321
#define BUFFER_LEN 1024

int main(int argc, char *argv[])
{
    int sockfd; /* descriptor a usar con el socket */
    struct sockaddr_in their_addr; /* almacenara la direccion IP y numero de puerto del servidor */
    struct hostent *he; /* para obtener nombre del host */
    int numbytes; /* conteo de bytes a escribir */
    if (argc != 3) {
        fprintf(stderr, "\nuso: %s cliente hostname mensaje\n", argv[0]);
        exit(1);
    }

    /* convertimos el hostname a su direccion IP */
    if ((he=gethostbyname(argv[1])) == NULL) {
        perror("gethostbyname");
        exit(1);
    }

    /* Creamos el socket */
    if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) == -1) {
        perror("socket");
        exit(2);
    }

    /* a donde mandar */
    their_addr.sin_family = AF_INET; /* usa host byte order */
    their_addr.sin_port = htons(SERVER_PORT); /* usa network byte order */
    their_addr.sin_addr = *((struct in_addr *)he->h_addr);
    bzero(&(their_addr.sin_zero), 8); /* pone en cero el resto */

    /* enviamos el mensaje */
    if ((numbytes=sendto(sockfd,argv[2],strlen(argv[2]),0,(struct sockaddr *)&their_addr,
sizeof(struct sockaddr))) == -1) {
        perror("sendto");
        exit(2);
    }

    printf("enviados %d bytes hacia %s\n",numbytes,inet_ntoa(their_addr.sin_addr));

    /* cierro socket */
    close(sockfd);
    exit (0);
}

```

3.- Compile y pruebe ambos programas. Deberá obtener alguna salida como la que se muestra abajo:

The screenshot shows a Linux desktop with a terminal window titled 'mtorrealba@nubia-laptop: ~/Escritorio'. The terminal displays a process list with columns for PID, PPID, USER, TIME, and COMMAND. Below the list, the user runs a command to send a UDP message to localhost, and the output shows the message was received and processed by the server program.

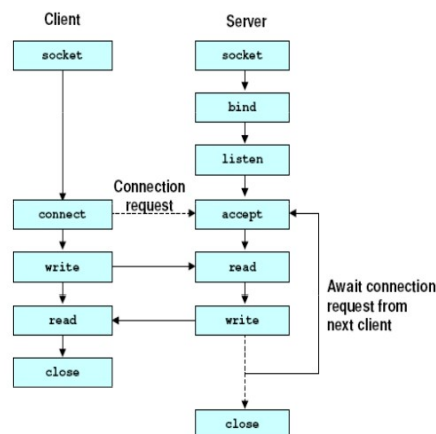
```

root      2864      2  0 18:02 ?        00:00:00 [kworker/1:1]
1001      2869      1  0 18:02 ?        00:00:00 /usr/lib/dconf/dconf-service
root      3065      1  0 18:09 ?        00:00:00 /usr/bin/python /usr/lib/system-
lp        3175    1247  0 18:10 ?        00:00:00 /usr/lib/cups/notifier/dbus dbus
root      3358      2  0 18:16 ?        00:00:00 [kworker/0:1]
root      3388      2  0 18:17 ?        00:00:00 [kworker/1:2]
1001      3569      1  0 18:26 ?        00:00:01 gnome-terminal
1001      3577    3569  0 18:26 ?        00:00:00 gnome-pty-helper
1001      3578    3569  0 18:26 pts/0    00:00:00 bash
root      3705      2  0 18:27 ?        00:00:00 [kworker/0:2]
root      3808      2  0 18:31 ?        00:00:00 [kworker/1:0]
root      3864      2  0 18:32 ?        00:00:00 [kworker/0:3]
root      3879    3578  0 18:33 pts/0    00:00:00 sudo ./server_udp
root      3880    3879  0 18:33 pts/0    00:00:00 ./server_udp
1001      3895    3578  0 18:34 pts/0    00:00:00 ps -ef
mtorrealba@nubia-laptop:~/Escritorio$ ./client_udp localhost "mensaje de prueba"
enviados 17 bytes hacia 127.0.0.1
paquete proveniente de : 127.0.0.1
longitud del paquete en bytes : 17
el paquete contiene : mensaje de prueba
[1]+  Hecho                  sudo ./server_udp
mtorrealba@nubia-laptop:~/Escritorio$

```

4.- Discuta las variaciones del ejemplo superior si se estuviese programando con TCP.

Esquema general de la comunicación Cliente / Servidor



5.- Si dispone de tiempo discuta la presentación sobre “gdb”.