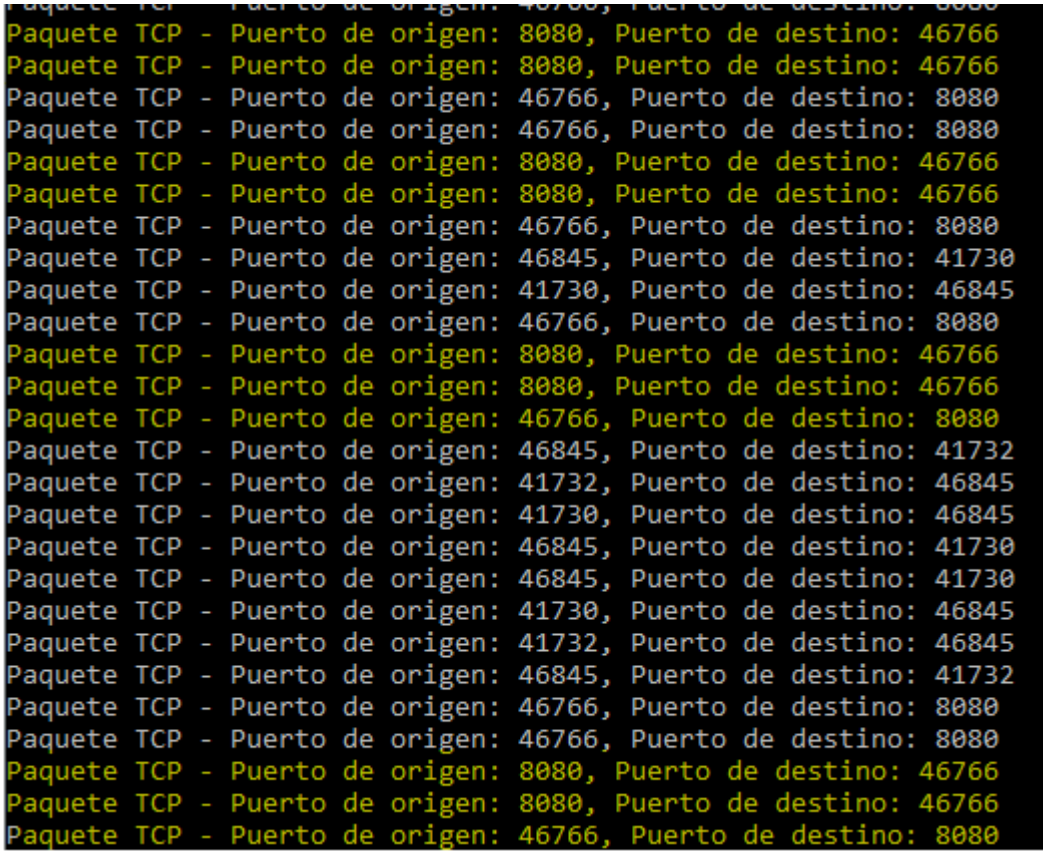
- Utilizando el código raw.c como base escribir un "sniffer" que es un programa que muestra el contenido del tráfico que llega.
  Utilizamos el raw_sniffer.c para poder mostrar el contenido del tráfico

```c
/*
Protocolos de Internet- Javier Ouret
RAW SOCKETS VERSION SIMPLIFICADA
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/socket.h>
#include <netinet/ip.h>
#include <netinet/udp.h>
#include <netinet/tcp.h>
#include <arpa/inet.h>

#define BUFFER_SIZE 65536

void procesar_paquete(unsigned char *buffer, int size) {
    struct iphdr *encabezado_ip = (struct iphdr *)buffer;
    unsigned short longitud_encabezado_ip = encabezado_ip->ihl * 4;

    if (encabezado_ip->protocol == IPPROTO_TCP) {
        struct tcphdr *encabezado_tcp = (struct tcphdr *)(buffer + longitud_encabezado_ip);
        unsigned int puerto_origen = ntohs(encabezado_tcp->source);
        unsigned int puerto_destino = ntohs(encabezado_tcp->dest);

        printf("Paquete TCP - Puerto de origen: %u, Puerto de destino: %u\n", puerto_origen, puerto_destino);
    } else if (encabezado_ip->protocol == IPPROTO_UDP) {
        struct udphdr *encabezado_udp = (struct udphdr *)(buffer + longitud_encabezado_ip);
        unsigned int puerto_origen = ntohs(encabezado_udp->source);
        unsigned int puerto_destino = ntohs(encabezado_udp->dest);

        printf("Paquete UDP - Puerto de origen: %u, Puerto de destino: %u\n", puerto_origen, puerto_destino);
    } else if (encabezado_ip->protocol == IPPROTO_ICMP) {
        printf("Paquete ICMP\n");
    } else {
        printf("Paquete de protocolo desconocido\n");
    }
}
```

```c
int main() {
    int sockfd;
    unsigned char buffer[BUFFER_SIZE];

    // Crear socket raw
    if ((sockfd = socket(AF_INET, SOCK_RAW, IPPROTO_TCP)) < 0) {
        perror("socket");
        exit(EXIT_FAILURE);
    }

    // Recibir paquetes
    while (1) {
        int bytes_recibidos = recvfrom(sockfd, buffer, sizeof(buffer), 0, NULL, NULL);
        if (bytes_recibidos < 0) {
            perror("recvfrom");
            exit(EXIT_FAILURE);
        }

        procesar_paquete(buffer, bytes_recibidos);
    }

    return 0;
}
```

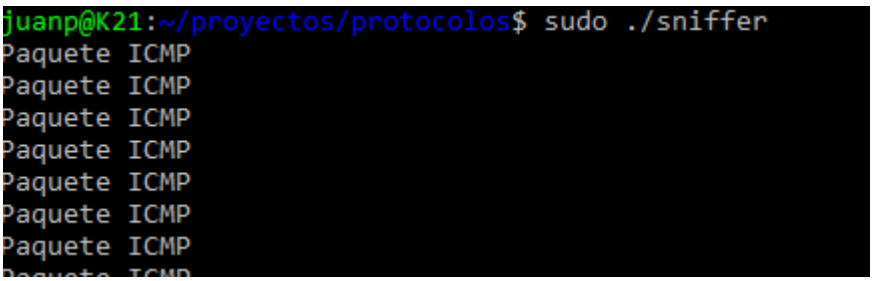
- Enviar tráfico al "sniffer" desde el cliente escrito en la parte A del TP1
  Para poder el ver las interacción C-S y ver su tráfico por el sniffer es simplemente ejecutar el sniffer.c, seguido por el servidor y el cliente y en base al puerto utilizado se puede ver el tráfico del mismo (En este caso el puerto del tráfico es 8080)



  En este caso el tráfico muestra la conexión C-S y su interacción concurrente como así las últimas tres líneas muestran la finalización de la comunicación de la misma

- Enviar tráfico ICMP al "sniffer" y mostrar los resultados del LOG con comentarios.
  Para que se pueda enviar un tráfico ICMP al sniffer es necesario cambiar “IPPROTO_TCP” a “IPPROTO_TCP” en el sniffer.c como en el cliente.c

  Hay dos formas para testear el muestreo de tráfico ICMP al sniffer. La primera sería, realizar un ping, un servicio usado por ICMP, o se modifica el socket de cliente para que acepte el protocolo de ip de tipo ICMP.



```
juanp@K21:~$ cd proyectos/protocolos
juanp@K21:~/proyectos/protocolos$ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.105 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.086 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.143 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.075 ms
^C
--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3130ms
rtt min/avg/max/mdev = 0.075/0.102/0.143/0.025 ms
```

Testeo de tráfico ICMP usando ping para mostrar el tráfico.

```
juanp@K21:~/proyectos/protocolos$ sudo ./sniffer
Paquete ICMP
Paquete ICMP
```

```
juanp@K21:~/proyectos/protocolos$ sudo ./cliente 127.0.0.1
Socket creado ..
Conectado al servidor..
Ingrese texto : hola
Servidor : EIngrese texto : hola
```

Testeo de tráfico ICMP usando el cliente para mostrar el tráfico.