TP 1-B

Integrantes:

- Cardinale Ignacio
- delValle Facundo
- Dimperio Bautista
- Ratcliffe Patricio.

1- Utilizando el código raw.c como base escribir un "sniffer" que es un programa que muestra el contenido del tráfico que llega.

Primero creamos el raw socket con el protocolo TCP:

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

Lo configuramos para que reciba todos los paquetes que le llegan:

```
50
51      // Recibir paquetes
52      while (1) {
53          int bytes_recibidos = recvfrom(sockfd, buffer, sizeof(buffer), 0, NULL, NULL);
54          if (bytes_recibidos < 0) {
55              perror("recvfrom");
56              exit(EXIT_FAILURE);
57          }
58
59          procesar_paquete(buffer, bytes_recibidos);
60      }
```

Después a cada paquete que llegaba lo llevamos a un función el cual captura el encabezado (las primeras dos líneas de código) y después compara este encabezado con el protocolo correspondiente, IPPROTO_TCP para TCP, IPPROTO_UDP para UDP, IPPROTO_ICMP para ICMP y dependiendo que tipo de paquete era que tenga una determinada salida:

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

    // Offset para los datos del paquete
    unsigned char *datos;
    int longitud_datos;

    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);

        // Calcular el offset de los datos del paquete TCP
        unsigned short longitud_encabezado_tcp = encabezado_tcp->doff * 4;
        datos = buffer + longitud_encabezado_ip + longitud_encabezado_tcp;
        longitud_datos = size - longitud_encabezado_ip - longitud_encabezado_tcp;

        // Imprimir los datos del paquete TCP
        printf("Datos del paquete TCP: ");
        for (int i = 0; i < longitud_datos; i++) {
            printf("%02x ", datos[i]);
        }
        printf("\n\n");
    } 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);

        // Calcular el offset de los datos del paquete UDP
        datos = buffer + longitud_encabezado_ip + sizeof(struct udphdr);
        longitud_datos = size - longitud_encabezado_ip - sizeof(struct udphdr);

        // Imprimir los datos del paquete UDP
        printf("Datos del paquete UDP: ");
        for (int i = 0; i < longitud_datos; i++) {
            printf("%02x ", datos[i]);
        }
        printf("\n\n");
    } else if (encabezado_ip->protocol == IPPROTO_ICMP) {
        // Calcular el offset de los datos del paquete ICMP
        datos = buffer + longitud_encabezado_ip;
```

```
} else if (encabezado_ip->protocol == IPPROTO_ICMP) {
    // Calcular el offset de los datos del paquete ICMP
    datos = buffer + longitud_encabezado_ip;
    longitud_datos = size - longitud_encabezado_ip;

    printf("Paquete ICMP\n");

    // Imprimir los datos del paquete ICMP
    printf("Datos del paquete ICMP: ");
    for (int i = 0; i < longitud_datos; i++) {
        printf("%02x ", datos[i]);
    }
    printf("\n\n");
}
else {
    printf("Paquete de protocolo desconocido\n");
}
```

En esta determinada salida no solo identifica el protocolo sino tambien imprime los datos que contiene el paquere, en este caso en hexadecimal.

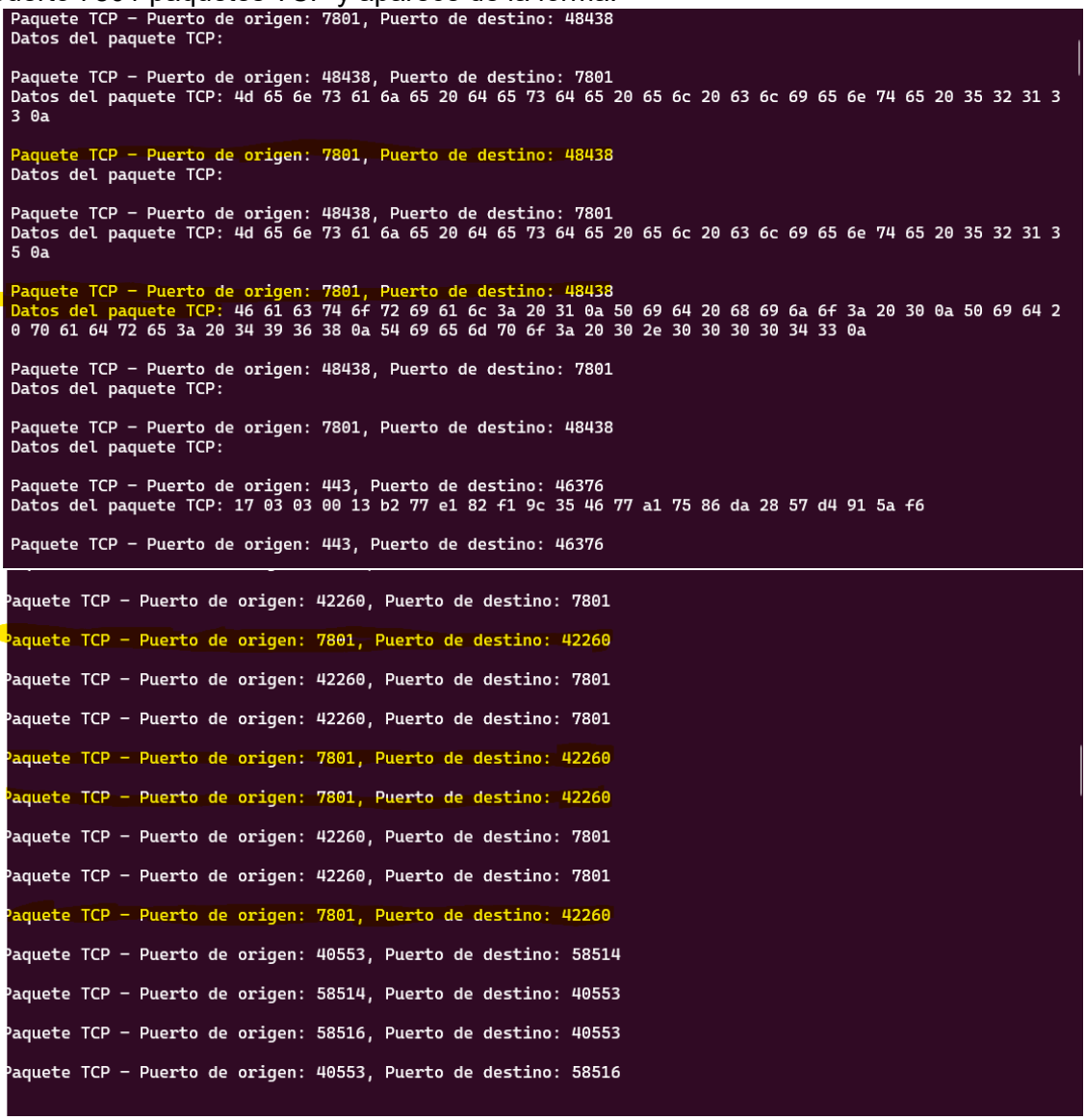
La salida al ejecutar el código es:

```
3 39 64 65 37 32 37 31 65 2f 76 73 63 6f 64 65 2e 6c 6f 63 6b 22 2c 22 73 63 68 65 6d 65 22 3a 22 76 73 63 6
f 64 65 2d 72 65 6d 6f 74 65 22 2c 22 61 75 74 68 6f 72 69 74 79 22 3a 22 77 73 6c 2b 75 62 75 6e 74 75 22 7
d 2c 22 74 79 70 65 22 3a 30 7d

Paquete TCP - Puerto de origen: 50344, Puerto de destino: 43527
Datos del paquete TCP:

Paquete TCP - Puerto de origen: 50350, Puerto de destino: 43527
Datos del paquete TCP: 82 fe 00 f6 10 58 cc 8b 11 58 cc 86 c3 58 cc 9a 64 58 cc 8b f9 59 cc 8b 17 55 99 87 3
4 37 a2 cd 79 34 a9 ce 66 3d a2 ff 10 58 cc 59 4b 23 ee e8 62 3d ad ff 75 3c ee b1 4b 05 e0 a9 73 30 ad e5 7
7 3d a8 a9 2a 03 b7 a9 34 35 a5 ef 32 62 fd a7 32 28 ad ff 78 7a f6 a9 3f 30 a3 e6 75 77 a2 ea 73 30 a3 a4 3
e 2e bf e8 7f 3c a9 a6 63 3d be fd 75 2a e3 ef 71 2c ad a4 45 2b a9 f9 3f 2f a3 f9 7b 2b bc ea 73 3d 9f ff 7
f 2a ad ec 75 77 ad bb 73 69 fb ee 75 3d ad b3 22 61 ae ef 27 6b ad b2 72 6e ff bf 71 3b f5 ef 75 6f fe bc 2
1 3d e3 fd 63 3b a3 ef 75 76 a0 e4 73 33 ee a7 32 2b af e3 75 35 a9 a9 2a 7a ba f8 73 37 a8 ee 3d 2a a9 e6 7
f 2c a9 a9 3c 7a ad fe 64 30 a3 f9 79 2c b5 a9 2a 7a bb f8 7c 73 b9 e9 65 36 b8 fe 32 25 91 a7 32 3c a9 e7 7
5 2c a9 ef 32 62 97 d6 6d 05

Paquete TCP - Puerto de origen: 43527, Puerto de destino: 50350
Datos del paquete TCP: 82 24 01 00 00 11 75 00 00 0d d3 00 00 00 05 05 00 00 07 0d 01 00 00 11 76 00 00 0d d
3 00 00 00 05 07 00 00 07 0d

Paquete TCP - Puerto de origen: 50350, Puerto de destino: 43527
Datos del paquete TCP:

Paquete TCP - Puerto de origen: 50350, Puerto de destino: 43527
Datos del paquete TCP: 82 8d e1 50 e0 a2 e8 50 e0 a2 e1 50 e0 b3 97 50 e0 a2 e1

Paquete TCP - Puerto de origen: 43527, Puerto de destino: 50350
```

```
Paquete TCP - Puerto de origen: 40553, Puerto de destino: 58514

Paquete TCP - Puerto de origen: 58514, Puerto de destino: 40553

Paquete TCP - Puerto de origen: 58514, Puerto de destino: 40553

Paquete TCP - Puerto de origen: 40553, Puerto de destino: 58514

Paquete TCP - Puerto de origen: 58516, Puerto de destino: 40553

Paquete TCP - Puerto de origen: 40553, Puerto de destino: 58516

Paquete TCP - Puerto de origen: 40553, Puerto de destino: 58514

Paquete TCP - Puerto de origen: 58514, Puerto de destino: 40553

Paquete TCP - Puerto de origen: 58516, Puerto de destino: 40553

Paquete TCP - Puerto de origen: 40553, Puerto de destino: 58516

Paquete TCP - Puerto de origen: 40553, Puerto de destino: 58516

Paquete TCP - Puerto de origen: 58516, Puerto de destino: 40553

Paquete TCP - Puerto de origen: 40553, Puerto de destino: 58516

Paquete TCP - Puerto de origen: 58516, Puerto de destino: 40553
```

2- Enviar tráfico al "sniffer" desde el cliente escrito en la parte A del TP1

Para esta parte usando el código del cliente de la parte A le mando por el puerto 7801 paquetes TCP y aparece de la forma:



3- Enviar tráfico ICMP al "sniffer" y mostrar los resultados del LOG con comentarios.

Para esta parte es igual a lo anterior lo único que cambia es el protocolo que vamos a usar para crear el socket, donde antes era:

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

Ahora es:

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

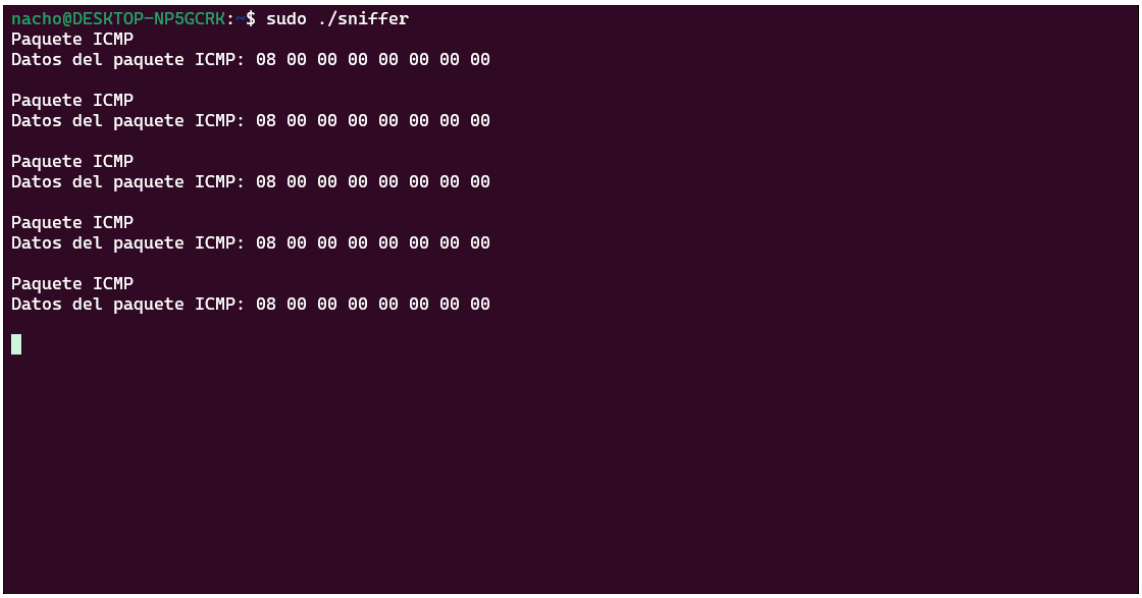
Y en el cliente al crear el socket se crea también con el protocolo ICMP:

```c
    // Crear socket raw ICMP

    if ((sockfd = socket(AF_INET, SOCK_RAW, IPPROTO_ICMP)) < 0) {
        perror("socket");
        exit(EXIT_FAILURE);
    }
```

Ahora en la consola cuando yo mando un paquete ICMP desde el cliente en la terminal de clienta aparece:

```
nacho@DESKTOP-NP5GCRK:~$ gcc -o cliente cliente_sniffer.c
nacho@DESKTOP-NP5GCRK:~$ sudo ./cliente 127.0.0.1
[sudo] password for nacho:
Paquete ICMP enviado al servidor 127.0.0.1
nacho@DESKTOP-NP5GCRK:~$ sudo ./cliente 127.0.0.1
Paquete ICMP enviado al servidor 127.0.0.1
nacho@DESKTOP-NP5GCRK:~$ sudo ./cliente 127.0.0.1
Paquete ICMP enviado al servidor 127.0.0.1
nacho@DESKTOP-NP5GCRK:~$ sudo ./cliente 127.0.0.1
Paquete ICMP enviado al servidor 127.0.0.1
nacho@DESKTOP-NP5GCRK:~$ sudo ./cliente 127.0.0.1
Paquete ICMP enviado al servidor 127.0.0.1
nacho@DESKTOP-NP5GCRK:~$ sudo ./cliente 127.0.0.1
Paquete ICMP enviado al servidor 127.0.0.1
nacho@DESKTOP-NP5GCRK:~$ sudo ./cliente 127.0.0.1
Paquete ICMP enviado al servidor 127.0.0.1
nacho@DESKTOP-NP5GCRK:~$ sudo ./cliente 127.0.0.1
Paquete ICMP enviado al servidor 127.0.0.1
nacho@DESKTOP-NP5GCRK:~$ sudo ./cliente 127.0.0.1
Paquete ICMP enviado al servidor 127.0.0.1
nacho@DESKTOP-NP5GCRK:~$
```

Y en la del servidor al llegar un paquete ICMP aparece como:


nacho@DESKTOP-NP5GCRK:~$ sudo ./sniffer
Paquete ICMP
Datos del paquete ICMP: 08 00 00 00 00 00 00 00
Paquete ICMP
Datos del paquete ICMP: 08 00 00 00 00 00 00 00
Paquete ICMP
Datos del paquete ICMP: 08 00 00 00 00 00 00 00
Paquete ICMP
Datos del paquete ICMP: 08 00 00 00 00 00 00 00
Paquete ICMP
Datos del paquete ICMP: 08 00 00 00 00 00 00 00