

## 1. SimpleSnif

La práctica consiste en la realización de una aplicación que permita obtener y procesar todo el tráfico que se obtiene a través de la tarjeta de red. Se trabajará en un entorno donde se dispone de una red Ethernet y donde está implementada la pila de protocolos TCP/IP. Dado que vamos a trabajar con protocolos TCP/IP, la realización de la práctica se hará utilizando la *interfaz socket*. Con esta práctica se pretende que el alumno se familiarice con la interfaz de sockets y sobre todo sepa identificar los formatos y los diferentes campos que componen los diferentes paquetes de información. En particular, a nivel de enlace se tratarán paquetes *ethernet*, a nivel de red paquetes *ip*, y a nivel de transporte paquetes *tcp*, *udp* e *icmp*.

La aplicación que se pretende realizar, se denominará *SimpleSnif* y debe permitir diferentes funcionalidades configurables en todo momento por el usuario. A continuación se detallan estas opciones:

- Opción **visionar/almacenar**: con esta opción se pretende que los paquetes procesados sean visionados por pantalla, almacenados en fichero o ambas a la vez.
- Opción **campos activos**: con esta opción se pretende poder indicar que campos del paquete deben ser visionados y/o almacenados. Los campos a tratar incluyen a todos los campos de las cabeceras ethernet, ip, tcp, udp e icmp así como el cuerpo del paquete. (Acceder a la dirección <http://www.etse.urv.es/~cmolina/XCI/practicas.html> para obtener más información sobre los formatos de los paquetes)
- Opción **filtrar paquetes**: con esta opción se pretende procesar paquetes que cumplan ciertos criterios. En el caso que ningún filtro esté activo, se procesarán todos los paquetes recibidos. Los filtros que deben ser implementados son los siguientes:
  - **Filtro 1**: se procesan los paquetes recibidos cuyas direcciones físicas (MAC address) se encuentren dentro de un listado.
  - **Filtro 2**: se procesan los paquetes recibidos cuyas direcciones lógicas (IP address) se encuentren dentro de un listado.
  - **Filtro 3**: se procesan los paquetes recibidos cuyos puertos se encuentren dentro de un listado.

Para facilitar el trabajo se proporciona una función que crea un socket y pone la tarjeta de red en modo promiscuo consiguiendo que el socket puede captar todo el tráfico que llega a la tarjeta. (Ver Anexo 1).

## 2. Requerimientos Mínimos

A continuación se detallan las funcionalidades mínimas que ha de proporcionar la infraestructura. Todos estos requerimientos son indispensables para aprobar la práctica

- No es necesario el uso de interfaz gráfica.
- Se utilizará el lenguaje de programación C.
- La infraestructura debe ser lo más robusta a fallos posible.
- Se deben implementar todas las funcionalidades descritas.

## 3. Notaciones

- Las prácticas se realizarán en grupos de dos personas como máximo.
- Se realizará una entrevista y prueba de la aplicación con todos los miembros del grupo.
- Mejoras y funcionalidades adicionales se tendrán en cuenta en la nota final.

## 4. Documentación a entregar.

- Disquete con los archivos fuentes y ejecutables. (libre de virus!!).
- Informe con las decisiones de diseño y los programas en alto nivel debidamente comentados.

## 5. Anexo 1

```

/*****
/*Parametros: nombre de la interficie, ej:eth0
/*Retorna: identificador de socket en modo promiscuo*/
*****/
int crear_socket_promiscuo (char *d)
{
    int fd;
    struct ifreq ifr;
    int s;

    fd=socket(AF_INET, SOCK_PACKET, htons(0x800));
    if (fd <0)
    {
        perror("cant get SOCK_PACKET socket");
        exit(0);
    }
    strcpy(ifr.ifr_name, d);
    s=ioctl(fd, SIOCGIFFLAGS, &ifr);
    if(s<0)
    {
        close(fd);
        perror("cant get flags");
        exit(0);
    }
    ifr.ifr_flags |= IFF_PROMISC;
    s=ioctl(fd, SIOCSIFFLAGS, &ifr);
    if (s<0) perror("cant set promiscuous mode");
    return fd;
}

```