PRACTICA 2

REDES 2

MARIO VALDEMARO GARCÍA ROQUÉ

ROBERTO GARCÍA TEODORO

GRUPO 2313

Desarrollo:

En esta práctica hemos tenido que implementar un servidor IRC

Creamos las siguientes estructuras para implementarlo:

Esta estructura representa a los clientes, con los diferentes campos que los componen:

```
typedef struct{
        char nick[9];
        char realName[REAL];
        char* userName;
        char* ip;
        char* id;
        int socket;
        bool registeredConnection;
        struct tm* lastConnection;
        pthread_t * hilo;
}CLIENT;
```

Esta estructura, channeluser relaciona los diferentes clientes que se encuentran en un canal:

```
typedef struct{
        CLIENT * client;
        bool op;
        bool voice;
} CHANNELUSER;
```

La estructura channel define los campos que debe tener un canal:

```
typedef struct{
        char* name;
        LinkedList * users;
        LinkedList * bannedUsers;
        LinkedList * invitedUsers;
        bool private;
        bool invite;
        bool voice;
        bool hiden;
        bool topicProtect;
}CHANNEL;
```

Ahora hablando de la parte realizada

No hemos tenido tiempo a implementar los comandos:

- Topic
- Names
- List

- Invite
- Kick
- Notice
- Away
- Die
- Users
- Wallops
- Userhost
- Ison
- Motd
- Lusers
- Versión
- Stats
- Connect
- Trace
- Admin
- Info
- Algunos mode

Pese a que si contásemos con más tiempo podríamos implementarlos con no mucha dificultad ya que una vez hemos comprendido, digamos, el proceso mediante el cual hemos ido realizando las diferentes implementaciones hemos ido desarrollando la capacidad para ir implementando los comandos.

Consideramos que lo que más nos costó fue el “arrancar”, una vez que fuimos rodando fuimos desarrollando más rápidamente.

Aunque si nos ha dado tiempo a implementar los siguientes:

- Envio y recepción de mensajes
- Conexión
- Desconexión
- Leave
- Part
- Quit
- Modes +i y +v, aunque no podemos invitar ni dar voz a gente, pero el modo +i puede usarse para hacer el canal privado para los miembros que ya estén. Tampoco podemos cambiar el modo de usuario.
- Nick
- Join
- Ping y pong
- Privmsg
- Error
- List

Tenemos una especie de bug en el mode, ya que si habilitamos 2 modos nos devuelve el siguiente mensaje:

" Canal #a modos: +is} ", recibimos ese '}', que es basura y falta un valor que debe ser hexadecimal que debería venir.

Los hilos que existen en el sistema son en total N+2 donde N es el número de clientes y 1 es un hilo adicional que es de servicio para atender las peticiones de llegada este es el hilo principal y otro es un hilo de mantenimiento de conexión este último hilo se encarga de manejar las el timeout de los usuarios de la siguiente forma:

Tenemos una función llamada "modulo60" que se encarga de ver cuánto tiempo ha pasado entre dos instantes de tiempo.

```
int modulo60(struct tm* t1, struct tm* t2){
	int auxt1=t1->tm_sec;
	int auxt2=t2->tm_sec;
	if(t1->tm_min > t2->tm_min){
		auxt1+=60;
	}
	if(auxt1-auxt2 > 40){
		return 2;
	}
	else if(auxt1-auxt2 > 20){
		return 1;
	}
	return 0;
}
```

El procedimiento funciona de la siguiente manera:
llega un momento nuevo y uno viejo,
si el minuto es diferente, se suma 60 a los segundos del viejo,
de forma que podemos saber el tiempo que ha pasado en segundo entre los dos instantes de tiempo.

Llamado de la siguiente forma en un switch podemos ver dos casos, si han pasado más de 20 segundos, mandamos un ping, y si han pasado más de 40, echamos al cliente por timeout:

```
switch(modulo60(tm, getLastConnectionClient(auxclient))){
	case 1:
		IRC_Ping(command, NULL, "Durruti", NULL);
		escribir(getSocketClient(auxclient), command);
	break;
	case 2:
		IRC_Quit(msg, getIdClient(auxclient), "timeout");
		broadcastMsg(msg,NULL, NULL);
		syslog (LOG_ERR, "usuario expulsado");
		borrarCliente(auxclient);
		pthread_cancel(*getPthreadClient(auxclient));
	break;
}
```

Por otro lado, debemos comentar el uso de la memoria, concretamente que no liberamos el uso que hacemos de la memoria, esto es debido a que intentamos avanzar en ese aspecto, liberando allá donde la usáramos, pero comenzó a darnos violaciones de segmento, y debido al escaso tiempo del que disponíamos, decidimos no liberar en ninguna parte, aunque lo dejamos como una opción para mejorar en un futuro.

También tenemos que mencionar un posible escaso uso de los semáforos, ya que comenzamos a poner (creemos que en exceso) y se nos producían interbloqueos, por lo que decidimos reducir el número de semáforos que teníamos, dejando también pendiente mejorar ese aspecto.

En conclusión, nos ha parecido una práctica muy interesante (ya que agradecemos poder hacer cosas “útiles”, es decir, que traspasen lo teórico viendo algo real y cómo nosotros podemos crearlo también.

Aunque la hubiéramos aprovechado mejor con más tiempo y menos carga de trabajo por parte de otras asignaturas.