

## Practica 1 - Redes de Comunicaciones II

Generado por Doxygen 1.8.9.1

Martes, 15 de Marzo de 2016 23:53:56



# Índice general

<b>1</b>	<b>Índice de archivos</b>	<b>1</b>
1.1	Lista de archivos . . . . .	1
<b>2</b>	<b>Documentación de archivos</b>	<b>3</b>
2.1	Referencia del Archivo src/G-2301-01-P1-irc_server.c . . . . .	3
2.1.1	Descripción detallada . . . . .	4
2.1.2	Documentación de las funciones . . . . .	4
2.1.2.1	get_motd_path . . . . .	4
2.1.2.2	get_nick . . . . .	4
2.1.2.3	get_socketd_bynick . . . . .	5
2.1.2.4	get_socketd_byuser . . . . .	5
2.1.2.5	get_user . . . . .	5
2.1.2.6	handler . . . . .	5
2.1.2.7	main . . . . .	6
2.1.2.8	process_command . . . . .	7
2.1.2.9	set_nick . . . . .	7
2.1.2.10	set_user . . . . .	7
2.1.2.11	timeout_thread . . . . .	7
2.1.2.12	update_timeout . . . . .	8
2.2	Referencia del Archivo src/lib/G-2301-01-P1-daemon.c . . . . .	8
2.2.1	Descripción detallada . . . . .	8
2.2.2	Documentación de las funciones . . . . .	8
2.2.2.1	daemonize . . . . .	8
2.3	Referencia del Archivo src/lib/G-2301-01-P1-irc.c . . . . .	9
2.3.1	Descripción detallada . . . . .	10
2.3.2	Documentación de las funciones . . . . .	10
2.3.2.1	away . . . . .	10
2.3.2.2	join . . . . .	10
2.3.2.3	kick . . . . .	10
2.3.2.4	list . . . . .	11
2.3.2.5	mode . . . . .	11

2.3.2.6	motd . . . . .	11
2.3.2.7	names . . . . .	11
2.3.2.8	nick . . . . .	12
2.3.2.9	no_command . . . . .	12
2.3.2.10	notice . . . . .	12
2.3.2.11	part . . . . .	12
2.3.2.12	ping . . . . .	13
2.3.2.13	pong . . . . .	13
2.3.2.14	privmsg . . . . .	13
2.3.2.15	quit . . . . .	13
2.3.2.16	time_c . . . . .	14
2.3.2.17	topic . . . . .	14
2.3.2.18	user . . . . .	14
2.3.2.19	version . . . . .	14
2.3.2.20	whois . . . . .	15
2.4	Referencia del Archivo src/lib/G-2301-01-P1-server.c . . . . .	15
2.4.1	Descripción detallada . . . . .	16
2.4.2	Documentación de las funciones . . . . .	16
2.4.2.1	connection_add . . . . .	16
2.4.2.2	connection_block . . . . .	16
2.4.2.3	connection_isblocked . . . . .	17
2.4.2.4	connection_rmv . . . . .	17
2.4.2.5	connection_unblock . . . . .	17
2.4.2.6	get_ip_from_connection . . . . .	17
2.4.2.7	is_connected . . . . .	18
2.4.2.8	is_readable . . . . .	19
2.4.2.9	server_launch . . . . .	19
2.4.2.10	server_stop . . . . .	19
2.4.2.11	set_do_on_disconnect . . . . .	19
2.4.2.12	sigint_handler . . . . .	19
2.5	Referencia del Archivo src/lib/G-2301-01-P1-tcp.c . . . . .	20
2.5.1	Descripción detallada . . . . .	20
2.5.2	Documentación de las funciones . . . . .	20
2.5.2.1	client_tcpsocket_open . . . . .	20
2.5.2.2	server_tcpsocket_open . . . . .	21
2.5.2.3	tcpsocket_accept . . . . .	21
2.5.2.4	tcpsocket_close . . . . .	21
2.5.2.5	tcpsocket_rcv . . . . .	21
2.5.2.6	tcpsocket_snd . . . . .	22
2.6	Referencia del Archivo src/lib/G-2301-01-P1-tools.c . . . . .	22

2.6.1	Descripción detallada . . . . .	22
2.6.2	Documentación de las funciones . . . . .	23
2.6.2.1	_strcmp . . . . .	23
2.6.2.2	_strlen . . . . .	23
2.7	Referencia del Archivo src/lib/G-2301-01-P1-udp.c . . . . .	23
2.7.1	Descripción detallada . . . . .	24
2.7.2	Documentación de las funciones . . . . .	24
2.7.2.1	client_udpsocket_open . . . . .	24
2.7.2.2	server_udpsocket_open . . . . .	24
2.7.2.3	udpsocket_rcv . . . . .	24
2.7.2.4	udpsocket_snd . . . . .	25
<b>Índice</b>		<b>27</b>



# Capítulo 1

## Indice de archivos

### 1.1. Lista de archivos

Lista de todos los archivos documentados y con descripciones breves:

src/ <a href="#">G-2301-01-P1-irc_server.c</a>	
Servidor IRC . . . . .	3
src/lib/ <a href="#">G-2301-01-P1-daemon.c</a>	
Libreria de creacion de un daemon . . . . .	8
src/lib/ <a href="#">G-2301-01-P1-irc.c</a>	
Libreria de manejo de mensajes IRC . . . . .	9
src/lib/ <a href="#">G-2301-01-P1-server.c</a>	
Libreria de manejo de un servidor de sockets . . . . .	15
src/lib/ <a href="#">G-2301-01-P1-tcp.c</a>	
Libreria de manejo de sockets TCP . . . . .	20
src/lib/ <a href="#">G-2301-01-P1-tools.c</a>	
Libreria de utilidades . . . . .	22
src/lib/ <a href="#">G-2301-01-P1-udp.c</a>	
Libreria de manejo de sockets UDP . . . . .	23





## Capítulo 2

# Documentación de archivos

### 2.1. Referencia del Archivo src/G-2301-01-P1-irc\_server.c

Servidor IRC.

```
#include "G-2301-01-P1-server.h"
#include "G-2301-01-P1-irc_server.h"
#include <G-2301-01-P1-irc.h>
#include <G-2301-01-P1-tools.h>
#include <G-2301-01-P1-daemon.h>
#include <redes2/irc.h>
#include <stdlib.h>
#include <stdio.h>
#include <syslog.h>
```

#### 'typedefs'

- typedef int(\* **comm\_t**) (char \*, void \*)

#### Funciones

- void **repair\_command** (char \*command)
- char \* **get\_motd\_path** ()  
*Devuelve el path del MOTD.*
- int **set\_user** (int socketd, char \***user**)  
*Asocia a un socketd el usuario.*
- int **set\_nick** (int socketd, char \***nick**)  
*Asocia a un socketd el nick.*
- int **get\_socketd\_byuser** (char \***user**)  
*Asocia a un socketd el nick.*
- int **get\_socketd\_bynick** (char \***nick**)  
*Busca el socket de un usuario por nick.*
- char \* **get\_nick** (int socketd)  
*Devuelve el nick de un usuario.*
- char \* **get\_user** (int socketd)  
*Devuelve el user de un usuario.*
- int **update\_timeout** (int socketd)  
*Actualiza el contador de timeout de un socketd.*

- void \* **handler** (void \*data)  
*Manejador de las conexiones.*
- void **do\_on\_disconnect** (void \*data)
- int **init\_commands** ()  
*Inicializa las llamadas de atencion de los comandos.*
- int **init\_memspace** ()  
*Inicializa las estructuras.*
- int **free\_memspace** ()  
*Borra las estructuras auxiliares del servidor.*
- int **process\_command** (char \*command, void \*data)  
*Procesa un comando.*
- void \* **timeout\_thread** (void \*data)  
*Hilo de desconexion por timeout.*
- int **read\_channels** ()
- int **main** (int argc, char \*\*argv)  
*Llamada principal del servidor.*

## Variables

- comm\_t **commands** [NCOMMANDS]
- char \*\* **users**
- char \*\* **nicks**
- char \* **timeout**
- char **running** =1
- char **path\_motd** [256] ="/tmp/irc\_tmp"
- pthread\_t **t\_timeout**
- pthread\_mutex\_t **m\_users**
- pthread\_mutex\_t **m\_nicks**
- pthread\_mutex\_t **m\_timeout**

### 2.1.1. Descripción detallada

Servidor IRC.

Autor

Sergio Fuentes [sergio.fuentesd@estudiante.uam.es](mailto:sergio.fuentesd@estudiante.uam.es)  
Daniel Perdices [daniel.perdices@estudiante.uam.es](mailto:daniel.perdices@estudiante.uam.es)

Fecha

2016/02/10

### 2.1.2. Documentación de las funciones

#### 2.1.2.1. char\* get\_motd\_path ( )

Devuelve el path del MOTD.

Devuelve

Dicho path al fichero

#### 2.1.2.2. char\* get\_nick ( int sockfd )

Devuelve el nick de un usuario.

## Parámetros

<i>socketd</i>	El socket del usuario
----------------	-----------------------

## Devuelve

El nick del usuario

2.1.2.3. `int get_socketd_bynick ( char * nick )`

Busca el socket de un usuario por nick.

## Parámetros

<i>nick</i>	Nick del usuario
-------------	------------------

## Devuelve

el socket del usuario

2.1.2.4. `int get_socketd_byuser ( char * user )`

Asocia a un socketd el nick.

## Parámetros

<i>socketd</i>	Socket del usuario
<i>user</i>	Nick del usuario

## Devuelve

IRCSVROK en caso adecuado, IRCSVERR (<0) en otro caso

2.1.2.5. `char* get_user ( int socketd )`

Devuelve el user de un usuario.

## Parámetros

<i>socketd</i>	El socket del usuario
----------------	-----------------------

## Devuelve

El user del usuario

2.1.2.6. `void* handler ( void * data )`

Manejador de las conexiones.

## Parámetros

<i>data</i>	Datos de la conexion y el mensaje TCP
-------------	---------------------------------------

## Devuelve

IRCSVR\_OK

2.1.2.7. `int main ( int argc, char ** argv )`

Llamada principal del servidor.

## Parámetros

<i>argc</i>	Num de argumentos
<i>argv</i>	Argumentos

## Devuelve

0

2.1.2.8. int process\_command ( char \* *command*, void \* *data* )

Procesa un comando.

## Parámetros

<i>command</i>	El comando
<i>data</i>	Datos de las conexion

## Devuelve

IRCSVROK en caso adecuado, IRCSVERR (&lt;0) en otro caso

2.1.2.9. int set\_nick ( int *socketd*, char \* *nick* )

Asocia a un socketd el nick.

## Parámetros

<i>socketd</i>	Socket del usuario
<i>user</i>	Nick del usuario

## Devuelve

IRCSVROK en caso adecuado, IRCSVERR (&lt;0) en otro caso

2.1.2.10. int set\_user ( int *socketd*, char \* *user* )

Asocia a un socketd el usuario.

## Parámetros

<i>socketd</i>	Socket del usuario
<i>user</i>	Nombre del usuario

## Devuelve

IRCSVROK en caso adecuado, IRCSVERR (&lt;0) en otro caso

2.1.2.11. void\* timeout\_thread ( void \* *data* )

Hilo de desconexion por timeout.

**Parámetros**

<i>data</i>	NULL
-------------	------

**Devuelve**

No devuelve ningun valor

**2.1.2.12. int update\_timeout ( int *socketd* )**

Actualiza el contador de timeout de un socketd.

**Parámetros**

<i>socketd</i>	El socket del usuario
----------------	-----------------------

**Devuelve**

IRCSVROK

## 2.2. Referencia del Archivo src/lib/G-2301-01-P1-daemon.c

Libreria de creacion de un daemon.

```
#include <stdio.h>
#include <stdlib.h>
#include <syslog.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
#include <string.h>
#include "G-2301-01-P1-daemon.h"
```

**Funciones**

- int [daemonize](#) (char \**svrcid*)  
*Crea un servicio daemon identificado.*

### 2.2.1. Descripción detallada

Libreria de creacion de un daemon.

**Autor**

Sergio Fuentes [sergio.fuentesd@estudiante.uam.es](mailto:sergio.fuentesd@estudiante.uam.es)  
Daniel Perdices [daniel.perdices@estudiante.uam.es](mailto:daniel.perdices@estudiante.uam.es)

**Fecha**

2016/02/10

### 2.2.2. Documentación de las funciones

**2.2.2.1. int daemonize ( char \* *svrcid* )**

Crea un servicio daemon identificado.

**Parámetros**

<i>svrcid</i>	Identificación del daemon
---------------	---------------------------

**Devuelve**

DMNOK o DMNERR\_\* en caso de error

## 2.3. Referencia del Archivo src/lib/G-2301-01-P1-irc.c

Librería de manejo de mensajes IRC.

```
#include <G-2301-01-P1-irc.h>
#include <redes2/irc.h>
#include <syslog.h>
#include <G-2301-01-P1-server.h>
#include <G-2301-01-P1-tools.h>
#include "G-2301-01-P1-irc_server.h"
```

**Funciones**

- int **nick** (char \*command, void \*more)  
*Atiende el comando NICK.*
- int **user** (char \*command, void \*more)  
*Atiende el comando USER.*
- int **list** (char \*command, void \*more)  
*Atiende el comando LIST.*
- int **ping** (char \*command, void \*more)  
*Atiende el comando PING.*
- int **pong** (char \*command, void \*more)  
*"Atiende" el comando PONG*
- int **join** (char \*command, void \*more)  
*Atiende el comando JOIN.*
- int **privmsg** (char \*command, void \*more)  
*Atiende el comando PRIVMSG.*
- int **mode** (char \*command, void \*more)  
*Atiende el comando MODE.*
- int **part** (char \*command, void \*more)  
*Atiende el comando PART.*
- int **names** (char \*command, void \*more)  
*Atiende el comando NAMES.*
- int **no\_command** (char \*command, void \*more)  
*Atiende comandos desconocidos.*
- int **topic** (char \*command, void \*more)  
*Atiende el comando TOPIC.*
- int **whois** (char \*command, void \*more)  
*Atiende el comando WHOIS.*
- int **quit** (char \*command, void \*more)  
*Atiende el comando QUIT.*
- int **motd** (char \*command, void \*more)  
*Atiende el comando MOTD.*

- int `away` (char \*command, void \*more)  
*Atiende el comando AWAY.*
- int `kick` (char \*command, void \*more)  
*Atiende el comando KICK.*
- int `time_c` (char \*command, void \*more)  
*Atiende el comando TIME.*
- int `version` (char \*command, void \*more)  
*Atiende el comando VERSION.*
- int `notice` (char \*command, void \*more)  
*Atiende el comando NOTICE.*

### 2.3.1. Descripción detallada

Librería de manejo de mensajes IRC.

Autor

Sergio Fuentes [sergio.fuentesd@estudiante.uam.es](mailto:sergio.fuentesd@estudiante.uam.es)  
Daniel Perdices [daniel.perdices@estudiante.uam.es](mailto:daniel.perdices@estudiante.uam.es)

Fecha

2016/02/10

### 2.3.2. Documentación de las funciones

#### 2.3.2.1. int away ( char \* command, void \* more )

Atiende el comando AWAY.

Parámetros

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

Devuelve

Ningún valor definido, la función controlan el error de manera interna

#### 2.3.2.2. int join ( char \* command, void \* more )

Atiende el comando JOIN.

Parámetros

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

Devuelve

Ningún valor definido, la función controlan el error de manera interna

#### 2.3.2.3. int kick ( char \* command, void \* more )

Atiende el comando KICK.



**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.4. int list ( char \* *command*, void \* *more* )**

Atiende el comando LIST.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.5. int mode ( char \* *command*, void \* *more* )**

Atiende el comando MODE.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.6. int motd ( char \* *command*, void \* *more* )**

Atiende el comando MOTD.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.7. int names ( char \* *command*, void \* *more* )**

Atiende el comando NAMES.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.8. int nick ( char \* *command*, void \* *more* )**

Atiende el comando NICK.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.9. int no\_command ( char \* *command*, void \* *more* )**

Atiende comandos desconocidos.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.10. int notice ( char \* *command*, void \* *more* )**

Atiende el comando NOTICE.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.11. int part ( char \* *command*, void \* *more* )**

Atiende el comando PART.

## Parámetros

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

## Devuelve

Ningún valor definido, la función controlan el error de manera interna

2.3.2.12. int ping ( char \* *command*, void \* *more* )

Atiende el comando PING.

## Parámetros

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

## Devuelve

Ningún valor definido, la función controlan el error de manera interna

2.3.2.13. int pong ( char \* *command*, void \* *more* )

"Atiende" el comando PONG

## Parámetros

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

## Devuelve

Ningún valor definido, la función controlan el error de manera interna

2.3.2.14. int privmsg ( char \* *command*, void \* *more* )

Atiende el comando PRIVMSG.

## Parámetros

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

## Devuelve

Ningún valor definido, la función controlan el error de manera interna

2.3.2.15. int quit ( char \* *command*, void \* *more* )

Atiende el comando QUIT.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.16. int time\_c ( char \* *command*, void \* *more* )**

Atiende el comando TIME.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.17. int topic ( char \* *command*, void \* *more* )**

Atiende el comando TOPIC.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.18. int user ( char \* *command*, void \* *more* )**

Atiende el comando USER.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.19. int version ( char \* *command*, void \* *more* )**

Atiende el comando VERSION.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.3.2.20. int whois ( char \* *command*, void \* *more* )**

Atiende el comando WHOIS.

**Parámetros**

<i>command</i>	El comando recibido
<i>more</i>	Puntero a estructura conn_data auxiliar

**Devuelve**

Ningún valor definido, la función controlan el error de manera interna

**2.4. Referencia del Archivo srclib/G-2301-01-P1-server.c**

Librería de manejo de un servidor de sockets.

```
#include <stdlib.h>
#include <stdio.h>
#include <pthread.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <sys/select.h>
#include <sys/time.h>
#include <unistd.h>
#include <strings.h>
#include <string.h>
#include <signal.h>
#include <arpa/inet.h>
#include <G-2301-01-P1-tcp.h>
#include "G-2301-01-P1-server.h"
```

**Funciones**

- int [connection\\_add](#) (int socketd)  
*Añade una conexión al conjunto del select.*
- int [connection\\_rmv](#) (int socketd)  
*Borra una conexión del conjunto del select.*
- int [is\\_connected](#) (int socketd)  
*Indica si hay una conexión abierta.*
- int [is\\_readable](#) (int socketd)  
*Indica si hay una conexión que requiere atención.*
- int [connection\\_block](#) (int socketd)  
*Bloquea una conexión.*

- int `connection_unblock` (int `socketd`)  
*Desbloquea una conexion.*
- int `connection_isblocked` (int `socketd`)  
*Comprueba si una conexion esta bloqueada.*
- char \* `get_ip_from_connection` (int `socketd`)  
*Devuelve la ip asociada a un socked.*
- void `sigint_handler` (int sig)  
*Manejador de la señal.*
- int `server_launch` (uint16\_t port, void \*(\*`handler`)(void \*), void \*more)  
*Lanza un servidor que atiende cada mensaje recibido con la funcion handler.*
- int `server_stop` ()  
*Para el servidor TCP.*
- int `set_do_on_disconnect` (void(\*`handler`)(void \*))  
*Para el servidor TCP.*

## Variables

- pthread\_mutex\_t `mutex_conn`
- fd\_set `connections`
- fd\_set `readable`
- fd\_set `blocked`
- int `loop`
- char `ips` [FD\_SETSIZE][16]
- void(\* `handler_disconnection` )(void \*) = NULL

### 2.4.1. Descripción detallada

Libreria de manejo de un servidor de sockets.

#### Autor

Sergio Fuentes [sergio.fuentesd@estudiante.uam.es](mailto:sergio.fuentesd@estudiante.uam.es)  
Daniel Perdices [daniel.perdices@estudiante.uam.es](mailto:daniel.perdices@estudiante.uam.es)

#### Fecha

2016/02/01

### 2.4.2. Documentación de las funciones

#### 2.4.2.1. int `connection_add` ( int `socketd` )

Añade una conexion al conjunto del select.

#### Parámetros

<code>socketd</code>	Descriptor del socket
----------------------	-----------------------

#### Devuelve

SERVOK

#### 2.4.2.2. int `connection_block` ( int `socketd` )

Bloquea una conexion.

## Parámetros

<i>socketd</i>	Descriptor del socket
----------------	-----------------------

## Devuelve

SERVOK

2.4.2.3. int connection\_isblocked ( int *socketd* )

Comprueba si una conexion esta bloqueada.

## Parámetros

<i>socketd</i>	Descriptor del socket
----------------	-----------------------

## Devuelve

0 si no esta bloqueada, otro valor si esta bloqueada

2.4.2.4. int connection\_rmv ( int *socketd* )

Borra una conexion del conjunto del select.

## Parámetros

<i>socketd</i>	Descriptor del socket
----------------	-----------------------

## Devuelve

SERVOK

2.4.2.5. int connection\_unblock ( int *socketd* )

Desbloquea una conexion.

## Parámetros

<i>socketd</i>	Descriptor del socket
----------------	-----------------------

## Devuelve

SERVOK

2.4.2.6. char\* get\_ip\_from\_connection ( int *socketd* )

Devuelve la ip asociada a un socked.

## Parámetros

<i>socketd</i>	Descriptor del socket
----------------	-----------------------

## Devuelve

La ip del socket en decimal

#### 2.4.2.7. `int is_connected ( int socketd )`

Indica si hay una conexión abierta.



## Parámetros

<i>socketd</i>	Descriptor del socket
----------------	-----------------------

## Devuelve

0 si esta cerrada, cualquier valor en otro caso

2.4.2.8. `int is_readable ( int socketd )`

Indica si hay una conexion que requiere atencion.

## Parámetros

<i>socketd</i>	Descriptor del socket
----------------	-----------------------

## Devuelve

0 si esta cerrada, cualquier valor en otro caso

2.4.2.9. `int server_launch ( uint16_t port, void (*)(void *) handler, void * more )`

Lanza un servidor que atiende cada mensaje recibido con la funcion handler.

## Parámetros

<i>port</i>	Puerto en el que abrir el servidor
<i>handler</i>	Rutina de atencion de los mensajes
<i>more</i>	Parametros adicionales que se necesiten en la rutina de atencion

## Devuelve

SERVOK en caso de que el servidor termine correctamente un numero negativo en caso de error

2.4.2.10. `int server_stop ( )`

Para el servidor TCP.

## Devuelve

SERVOK si el servidor se para, SERVERR\_NRUN si no hay servidor

2.4.2.11. `int set_do_on_disconnect ( void (*)(void *) handler )`

Para el servidor TCP.

## Devuelve

SERVOK si el servidor se para, SERVERR\_NRUN si no hay servidor

2.4.2.12. `void sigint_handler ( int sig )`

Manejador de la señal.

## Parámetros

<i>sig</i>	señal recibido
------------	----------------

## Devuelve

0 si no esta bloqueada, otro valor si esta bloqueada

## 2.5. Referencia del Archivo src/lib/G-2301-01-P1-tcp.c

Libreria de manejo de sockets TCP.

```
#include "G-2301-01-P1-tcp.h"
#include <stdio.h>
#include <stdlib.h>
#include <netdb.h>
#include <pthread.h>
#include <strings.h>
#include <string.h>
#include <unistd.h>
#include <syslog.h>
```

## Funciones

- int [server\\_tcpsocket\\_open](#) (uint16\_t port, int \*socketd)  
*Abre un socket TCP para servidor.*
- int [client\\_tcpsocket\\_open](#) (uint16\_t port, int \*socketd, char \*hostname)  
*Abre un socket TCP para cliente.*
- int [tcpsocket\\_accept](#) (int socketd, tcpsocket\_args \*args)  
*Acepta una conexion a socket TCP.*
- int [tcpsocket\\_snd](#) (int socketd, void \*data, size\_t len)  
*Envia los datos a traves del socket TCP.*
- int [tcpsocket\\_rcv](#) (int socketd, void \*data, size\_t max, size\_t \*len)  
*Recibe a traves del socket TCP.*
- void [tcpsocket\\_close](#) (int socketd)  
*Cierra un socket TCP.*

### 2.5.1. Descripción detallada

Libreria de manejo de sockets TCP.

## Autor

Sergio Fuentes [sergio.fuentesd@estudiante.uam.es](mailto:sergio.fuentesd@estudiante.uam.es)  
Daniel Perdices [daniel.perdices@estudiante.uam.es](mailto:daniel.perdices@estudiante.uam.es)

## Fecha

2016/02/01

### 2.5.2. Documentación de las funciones

#### 2.5.2.1. int client\_tcpsocket\_open ( uint16\_t port, int \* socketd, char \* hostname )

Abre un socket TCP para cliente.

## Parámetros

<i>port</i>	Puerto desde el que se desea escuchar
<i>socketd</i>	Puntero al descriptor del socket

## Devuelve

TCPOK si todo fue correcto TCPERR\_ARGS/SOCKET/BIND/LISTEN en caso de error con estas funciones

2.5.2.2. `int server_tcpsocket_open ( uint16_t port, int * socketd )`

Abre un socket TCP para servidor.

## Parámetros

<i>port</i>	Puerto desde el que se desea escuchar
<i>socketd</i>	Puntero al descriptor del socket

## Devuelve

TCPOK si todo funciono correctamente TCPERR\_ARGS/SOCKET/BIND/LISTEN en caso de error con estas funciones

2.5.2.3. `int tcpsocket_accept ( int socketd, tcpsocket_args * args )`

Acepta una conexion a socket TCP.

## Parámetros

<i>socketd</i>	Descriptor del socket escuchado
<i>args</i>	estructura de datos de la conexion

## Devuelve

TCPOK si todo fue correcto TCPERR\_ARGS/ACCEPT/PTHREAD en caso de error con estas funciones

2.5.2.4. `void tcpsocket_close ( int socketd )`

Cierra un socket TCP.

## Parámetros

<i>socketd</i>	descriptor del socket
----------------	-----------------------

2.5.2.5. `int tcpsocket_rcv ( int socketd, void * data, size_t max, size_t * len )`

Recibe a traves del socket TCP.

## Parámetros

<i>socketd</i>	Descriptor del socket escuchado
<i>data</i>	puntero a donde se almacenan los datos

<i>max</i>	tamaño de la zona de memoria
<i>len</i>	puntero a la variable de longitud recibida

**Devuelve**

TCPOK si todo fue correcto TCPERR\_ARGS/RECV en caso de error con estas funciones

**2.5.2.6. int tcpsocket\_snd ( int *socketd*, void \* *data*, size\_t *len* )**

Envia los datos a traves del socket TCP.

**Parámetros**

<i>socketd</i>	Descriptor del socket escuchado
<i>data</i>	puntero a los datos que se han de enviar
<i>len</i>	tamano de los datos a enviar

**Devuelve**

TCPOK si todo fue correcto TCPERR\_ARGS/SEND en caso de error con estas funciones

**2.6. Referencia del Archivo srclib/G-2301-01-P1-tools.c**

Libreria de utilidades.

```
#include "G-2301-01-P1-tcp.h"
#include <stdio.h>
#include <stdlib.h>
#include <netdb.h>
#include <pthread.h>
#include <strings.h>
#include <string.h>
#include <unistd.h>
#include <syslog.h>
```

**'defines'**

- #define **strlen** *\_strlen*
- #define **strcmp** *\_strcmp*

**Funciones**

- size\_t *\_strlen* (const char \*s)  
*Funcion strlen de strings.h.*
- int *\_strcmp* (const char \*a, const char \*b)  
*Funcion strcmp de strings.h.*

**2.6.1. Descripción detallada**

Libreria de utilidades.

**Autor**

Sergio Fuentes [sergio.fuentesd@estudiante.uam.es](mailto:sergio.fuentesd@estudiante.uam.es)  
Daniel Perdices [daniel.perdices@estudiante.uam.es](mailto:daniel.perdices@estudiante.uam.es)

**Fecha**

2016/02/01

**2.6.2. Documentación de las funciones****2.6.2.1. `int _strcmp ( const char * a, const char * b )`**

Funcion strcmp de strings.h.

**Parámetros**

<i>a</i>	Puntero a la cadena de caracteres
<i>b</i>	Puntero a la cadena de caracteres

**Devuelve**

la diferencia entre ambas cadenas

**2.6.2.2. `size_t _strlen ( const char * s )`**

Funcion strlen de strings.h.

**Parámetros**

<i>s</i>	Puntero a la cadena de caracteres
----------	-----------------------------------

**Devuelve**

la longitud

**2.7. Referencia del Archivo src/lib/G-2301-01-P1-udp.c**

Libreria de manejo de sockets UDP.

```
#include <G-2301-01-P1-udp.h>
```

**Funciones**

- `int server_udpsocket_open (uint16_t port, int *socketd)`  
*Abre un socket UDP para servidor.*
- `int client_udpsocket_open (uint16_t port, int *socketd)`  
*Abre un socket UDP para cliente.*
- `int udpsocket_snd (int socketd, void *data, size_t len, udpsocket_side *dst)`  
*Envia los datos a traves del socket UDP.*
- `int udpsocket_rcv (int socketd, void *data, size_t max, size_t *len, udpsocket_side *src)`  
*Recibe a traves del socket UDP.*

### 2.7.1. Descripción detallada

Librería de manejo de sockets UDP.

#### Autor

Sergio Fuentes [sergio.fuentesd@estudiante.uam.es](mailto:sergio.fuentesd@estudiante.uam.es)  
Daniel Perdices [daniel.perdices@estudiante.uam.es](mailto:daniel.perdices@estudiante.uam.es)

#### Fecha

2016/02/01

### 2.7.2. Documentación de las funciones

#### 2.7.2.1. `int client_udpsocket_open ( uint16_t port, int * socketd )`

Abre un socket UDP para cliente.

##### Parámetros

<i>port</i>	Puerto desde el que se desea escuchar
<i>socketd</i>	Puntero al descriptor del socket

##### Devuelve

UDPOK si todo fue correcto UDPERR\_ARGS/SOCKET/BIND/LISTEN en caso de error con estas funciones

#### 2.7.2.2. `int server_udpsocket_open ( uint16_t port, int * socketd )`

Abre un socket UDP para servidor.

##### Parámetros

<i>port</i>	Puerto desde el que se desea escuchar
<i>socketd</i>	Puntero al descriptor del socket

##### Devuelve

UDPOK si todo funciona correctamente UDPERR\_ARGS/SOCKET/BIND en caso de error con estas funciones

#### 2.7.2.3. `int udpsocket_rcv ( int socketd, void * data, size_t max, size_t * len, udpsocket_side * src )`

Recibe a través del socket UDP.

##### Parámetros

<i>socketd</i>	Descriptor del socket escuchado
<i>data</i>	puntero a donde se almacenan los datos
<i>max</i>	tamaño de la zona de memoria
<i>len</i>	puntero a la variable de longitud recibida

##### Devuelve

UDPOK si todo fue correcto UDPERR\_ARGS/RECV en caso de error con estas funciones

2.7.2.4. `int udpsocket_snd ( int socketd, void * data, size_t len, udpsocket_side * dst )`

Envia los datos a traves del socket UDP.

**Parámetros**

<i>socketd</i>	Descriptor del socket escuchado
<i>data</i>	puntero a los datos que se han de enviar
<i>len</i>	tamano de los datos a enviar

**Devuelve**

UDPOK si todo fue correcto UDPERR\_ARGS/SEND en caso de error con estas funciones



# Índice alfabético

- [\\_strcmp](#)  
G-2301-01-P1-tools.c, [23](#)
  - [\\_strlen](#)  
G-2301-01-P1-tools.c, [23](#)
- [away](#)  
G-2301-01-P1-irc.c, [10](#)
- [client\\_tcpsocket\\_open](#)  
G-2301-01-P1-tcp.c, [20](#)
- [client\\_udpsocket\\_open](#)  
G-2301-01-P1-udp.c, [24](#)
- [connection\\_add](#)  
G-2301-01-P1-server.c, [16](#)
- [connection\\_block](#)  
G-2301-01-P1-server.c, [16](#)
- [connection\\_isblocked](#)  
G-2301-01-P1-server.c, [17](#)
- [connection\\_rmv](#)  
G-2301-01-P1-server.c, [17](#)
- [connection\\_unblock](#)  
G-2301-01-P1-server.c, [17](#)
- [daemonize](#)  
G-2301-01-P1-daemon.c, [8](#)
- [G-2301-01-P1-daemon.c](#)  
  [daemonize](#), [8](#)
- [G-2301-01-P1-irc.c](#)  
  [away](#), [10](#)  
  [join](#), [10](#)  
  [kick](#), [10](#)  
  [list](#), [11](#)  
  [mode](#), [11](#)  
  [motd](#), [11](#)  
  [names](#), [11](#)  
  [nick](#), [12](#)  
  [no\\_command](#), [12](#)  
  [notice](#), [12](#)  
  [part](#), [12](#)  
  [ping](#), [13](#)  
  [pong](#), [13](#)  
  [privmsg](#), [13](#)  
  [quit](#), [13](#)  
  [time\\_c](#), [14](#)  
  [topic](#), [14](#)  
  [user](#), [14](#)  
  [version](#), [14](#)  
  [whois](#), [15](#)
- [G-2301-01-P1-irc\\_server.c](#)  
  [get\\_motd\\_path](#), [4](#)  
  [get\\_nick](#), [4](#)  
  [get\\_socketd\\_bynick](#), [5](#)  
  [get\\_socketd\\_byuser](#), [5](#)  
  [get\\_user](#), [5](#)  
  [handler](#), [5](#)  
  [main](#), [5](#)  
  [process\\_command](#), [7](#)  
  [set\\_nick](#), [7](#)  
  [set\\_user](#), [7](#)  
  [timeout\\_thread](#), [7](#)  
  [update\\_timeout](#), [8](#)
- [G-2301-01-P1-server.c](#)  
  [connection\\_add](#), [16](#)  
  [connection\\_block](#), [16](#)  
  [connection\\_isblocked](#), [17](#)  
  [connection\\_rmv](#), [17](#)  
  [connection\\_unblock](#), [17](#)  
  [get\\_ip\\_from\\_connection](#), [17](#)  
  [is\\_connected](#), [17](#)  
  [is\\_readable](#), [19](#)  
  [server\\_launch](#), [19](#)  
  [server\\_stop](#), [19](#)  
  [set\\_do\\_on\\_disconnect](#), [19](#)  
  [sigint\\_handler](#), [19](#)
- [G-2301-01-P1-tcp.c](#)  
  [client\\_tcpsocket\\_open](#), [20](#)  
  [server\\_tcpsocket\\_open](#), [21](#)  
  [tcpsocket\\_accept](#), [21](#)  
  [tcpsocket\\_close](#), [21](#)  
  [tcpsocket\\_rcv](#), [21](#)  
  [tcpsocket\\_snd](#), [22](#)
- [G-2301-01-P1-tools.c](#)  
  [\\_strcmp](#), [23](#)  
  [\\_strlen](#), [23](#)
- [G-2301-01-P1-udp.c](#)  
  [client\\_udpsocket\\_open](#), [24](#)  
  [server\\_udpsocket\\_open](#), [24](#)  
  [udpsocket\\_rcv](#), [24](#)  
  [udpsocket\\_snd](#), [24](#)
- [get\\_ip\\_from\\_connection](#)  
  G-2301-01-P1-server.c, [17](#)
- [get\\_motd\\_path](#)  
  G-2301-01-P1-irc\_server.c, [4](#)
- [get\\_nick](#)  
  G-2301-01-P1-irc\_server.c, [4](#)
- [get\\_socketd\\_bynick](#)  
  G-2301-01-P1-irc\_server.c, [5](#)
- [get\\_socketd\\_byuser](#)

- G-2301-01-P1-irc\_server.c, [5](#)
- get\_user
  - G-2301-01-P1-irc\_server.c, [5](#)
- handler
  - G-2301-01-P1-irc\_server.c, [5](#)
- is\_connected
  - G-2301-01-P1-server.c, [17](#)
- is\_readable
  - G-2301-01-P1-server.c, [19](#)
- join
  - G-2301-01-P1-irc.c, [10](#)
- kick
  - G-2301-01-P1-irc.c, [10](#)
- list
  - G-2301-01-P1-irc.c, [11](#)
- main
  - G-2301-01-P1-irc\_server.c, [5](#)
- mode
  - G-2301-01-P1-irc.c, [11](#)
- motd
  - G-2301-01-P1-irc.c, [11](#)
- names
  - G-2301-01-P1-irc.c, [11](#)
- nick
  - G-2301-01-P1-irc.c, [12](#)
- no\_command
  - G-2301-01-P1-irc.c, [12](#)
- notice
  - G-2301-01-P1-irc.c, [12](#)
- part
  - G-2301-01-P1-irc.c, [12](#)
- ping
  - G-2301-01-P1-irc.c, [13](#)
- pong
  - G-2301-01-P1-irc.c, [13](#)
- privmsg
  - G-2301-01-P1-irc.c, [13](#)
- process\_command
  - G-2301-01-P1-irc\_server.c, [7](#)
- quit
  - G-2301-01-P1-irc.c, [13](#)
- server\_launch
  - G-2301-01-P1-server.c, [19](#)
- server\_stop
  - G-2301-01-P1-server.c, [19](#)
- server\_tcpsocket\_open
  - G-2301-01-P1-tcp.c, [21](#)
- server\_udpsocket\_open
  - G-2301-01-P1-udp.c, [24](#)
- set\_do\_on\_disconnect
  - G-2301-01-P1-server.c, [19](#)
- set\_nick
  - G-2301-01-P1-irc\_server.c, [7](#)
- set\_user
  - G-2301-01-P1-irc\_server.c, [7](#)
- sigint\_handler
  - G-2301-01-P1-server.c, [19](#)
- src/G-2301-01-P1-irc\_server.c, [3](#)
- src/lib/G-2301-01-P1-daemon.c, [8](#)
- src/lib/G-2301-01-P1-irc.c, [9](#)
- src/lib/G-2301-01-P1-server.c, [15](#)
- src/lib/G-2301-01-P1-tcp.c, [20](#)
- src/lib/G-2301-01-P1-tools.c, [22](#)
- src/lib/G-2301-01-P1-udp.c, [23](#)
- tcpsocket\_accept
  - G-2301-01-P1-tcp.c, [21](#)
- tcpsocket\_close
  - G-2301-01-P1-tcp.c, [21](#)
- tcpsocket\_rcv
  - G-2301-01-P1-tcp.c, [21](#)
- tcpsocket\_snd
  - G-2301-01-P1-tcp.c, [22](#)
- time\_c
  - G-2301-01-P1-irc.c, [14](#)
- timeout\_thread
  - G-2301-01-P1-irc\_server.c, [7](#)
- topic
  - G-2301-01-P1-irc.c, [14](#)
- udpsocket\_rcv
  - G-2301-01-P1-udp.c, [24](#)
- udpsocket\_snd
  - G-2301-01-P1-udp.c, [24](#)
- update\_timeout
  - G-2301-01-P1-irc\_server.c, [8](#)
- user
  - G-2301-01-P1-irc.c, [14](#)
- version
  - G-2301-01-P1-irc.c, [14](#)
- whois
  - G-2301-01-P1-irc.c, [15](#)