**Chat Protocol**

**Introduction**

This is a network protocol specification document for a simple real-time chat communication protocol. The document specifies the network messages understood by servers and client using this communication protocol.

**Connection**

**Connect**

**Command:** CONNECT

**Parameters:** [username]

**Example payload:** CONNECT Ashleigh\_Smith

**Description:** Sends a connect request to a server. Once the client has a socket bound to the IP address and port of a server, it can send this command to fully initiate the session. The server responds with a packet containing one of the responses below. If the username is not registered, an OK packet is sent and the client can then join channels and send further packets. If the username is registered, the server will send a REQ\_AUTH response to the client, which must be answered with an AUTH packet by the client. Errors for server being unavailable, full or usernames being banned or taken may also be sent by the server.

**Responses:** OK, REQ\_AUTH, ERR\_UNAVAILABLE, ERR\_SERVERFULL, ERR\_USEREXISTS ERR\_PERMABANNED, ERR\_TEMPBANNED

**Disconnect**

**Command:** DISCONNECT

**Parameters:** None

**Description:** Disconnect from the server.

**Responses:** OK

**Ping (todo)**

**Command:** PING

**Parameters:** None

**Example payload:** PING

**Description:** Polls clients for connection status, to check if they are connected.

**Responses:** OK

**Users**

**Username rules**

Usernames should be unique and any client that supplies a username already in use will receive a ERR\_USEREXISTS packet as a response to their CONNECT packet.

Usernames must be a single alphanumeric string with no spaces. The maximum length of the username is 14 characters. Underscores are allowed to separate words (e.g. Ashleigh\_Smith).

The following are examples of valid characters:

* 0-9
* Aa-Zz
* \_
* [
* ]

**Register new user**

**Command:** REGISTER

**Parameters:** [username] [password]

**Example payload:** REGISTER Ashleigh a\_secure\_password

**Description:** When a client first connects, the server should check if the username supplied is registered. If it isn't, it can then be registered with a REGISTER packet. The parameters supplied should be a non-registered username and a password. There are currently no restrictions on password length.

**Responses:** OK

**Authenticate**

**Command:** AUTH

**Parameters:** [username] [password]

**Example payload:** AUTH Ashleigh a\_secure\_password

**Description:** If the server finds that the username supplied in the CONNECT packet is registered, it will send a REQ\_AUTH as a response to the client. The client must respond using this AUTH packet, supplying the username and the correct password. If the credentials are incorrect, the response will be ERR\_AUTH. If the user is not found, ERR\_NOSUCHUSER is returned. Note that for security reasons, the server should never specify which part of the credentials are wrong (i.e. username or password) – a generic ERR\_AUTH should be sent.

**Responses:** OK, ERR\_NOSUCHUSER, ERR\_AUTH

**Change password**

**Command:** USER\_PASS

**Parameters:** [username] [current\_password] [new\_password]

**Example payload:** USER\_PASS a\_poor\_password a\_secure\_password

**Description:** A client can use this packet to change the password of the currently connected user. The current user’s password must be supplied along with the new password. If the current password is incorrect, an ERR\_AUTH is sent to the client.

**Responses:** OK, ERR\_AUTH

**Admin system**

This should be described by adding "ERR\_PERMS" to the list of responses in commands that would require special privilege.

Examples: Kick, ban, mute, server restart, server shutdown, changing channel topic, etc

**Channels**

**Channel names**

Channel names can only be alphanumeric and must not contain any spaces. As a convention, channel names start with a “#” symbol. The character limit for channel names is 20 characters. Dashes or underscores may be used for channel names with multiple words (e.g. #super-secret-room).

**Join or leave a channel**

**Command:** CH\_JOIN

**Command:** CH\_LEAVE

**Parameters:** [channel]

**Example payload:** CH\_JOIN #foo

**Description:** Clients use these packets to join and leave channels. The channel name must be specified in both commands. Multiple channel names cannot be supplied.

**Responses:** OK

**List existing channels**

**Command:** CH\_LIST

**Parameters:** (public/private)

**Example payload:** CH\_LIST

**Description:** When the server receives this message, it will send a list of existing channels to the client. The optional parameter specifies whether to list *only* public or private channels. If this packet came from a non-admin user or a user who cannot see certain private channels, the server will only send the channels that the user has access to. If the user is an admin, all channels will be sent back to the client. The same behaviour happens when public or private is supplied as a non-admin user. The response is sent as a simple array. This array can be empty if no visible channels are found.

**Example response:** ["#foo","#bar","#super-secret-chat"]

**Listing users of channels**

**Command:** CH\_USERLIST

**Parameters:** [channel]

**Example payload:** CH\_USERLIST #foo

**Description:** Returns a list of users currently in a channel to the client. The response is sent as an array, which can be empty if no users are in the channel. Additionally, the server may send a “ERR\_NOSUCHCHANNEL” if the channel name is not found.

**Example responses:** [“Ashleigh\_Smith”, “Phixyn”], ERR\_NOSUCHCHANNEL

**Creating a channel**

**Command:** CH\_CREATE

**Parameters:** [name]

**Payload:** CH\_CREATE #foo

**Description:** Create a new channel with specified name. If the name is above 20 characters in length, or if invalid characters are found, an ERR\_CHANNELNAME is sent.

**Responses:** OK, ERR\_CHANNELNAME

**Deleting a channel**

**Command:** CH\_DESTROY

**Parameters:** [name]

**Payload:** CH\_DESTROY #foo

**Description:** Delete an existing channel with the specified name. Can only be used by an admin. If the user is not an admin, ERR\_NOTADMIN is returned. An ERR\_NOSUCHCHANNEL will be sent if the channel name specified is not found.

**Responses:** OK, ERR\_NOSUCHCHANNEL, ERR\_NOTADMIN

**Channel topic**

**Command:** CH\_TOPIC

**Parameters:** [channel] (message)

**Payload:** CH\_TOPIC A channel for general discussion

**Description:** Read or set the topic for the specified channel. This requires admin privileges.

**Responses:** OK, ERR\_NOSUCHCHANNEL, ERR\_NOTADMIN

**Messages**

**Command:** MSG

**Parameters:** [channel/user]

**Payload:** MSG Ashleigh Don't forget the milk!

MSG #general Does anyone else want to go home and play FFXIV?

**Description:** Send a message to an existing channel or a connected user

Public messages from a client are broadcast to all connected clients (except its origin) in the channel.

Private messages are only broadcast to the clients they are intended for, as opposed to all connected clients in a channel. Private messages are only sent to one client at a time.

**Responses:** OK

**Other**

**MOTD**

**Command:** MOTD

**Parameters:** (message)

**Payload:** MOTD Welcome to Zion's chat system.

**Description:** Return or set current Message Of The Day on the server. (server wide)

**Responses:** (Current MOTD message) (+ OK?), NO\_MOTD???

**Uptime**

**Command:** UPTIME

**Parameters:** None

**Payload:** UPTIME

**Description:** Return the server's current uptime.

**Responses:** UPTIME string (e.g. 7d, 11:03:02), OK (?), ERR\_PERMS (?)

**Appendix**

**Errors codes/messages**

**Common error cases**

Trying to send messages before authenticating

No permissions for certain actions (admin system)

ERR\_AUTH

ERR\_NOTADMIN

**Registration/Authentication errors**

ERR\_NOSUCHUSER

USER\_EXISTS

ERR\_PERMABANNED

ERR\_TEMPBANNED

USER\_INVALIDNAME

ERR\_UNAVAILABLE

ERR\_SERVERFULL

**References**