

IRC HACKS™

100 Industrial-Strength Tips & Tools



O'REILLY®

Paul Mutton

HACK
#31

IRC Over Telnet

Make a raw connection to an IRC server and spend a little time trying out the IRC protocol commands you'll be using in your program directly.

Jarkko Oikarinen first introduced IRC to the world in 1988. Five years later, he clearly defined the IRC protocol in RFC 1459, which made the whole protocol much more accessible. Armed with this information, you can get a better understanding of this simple text-based protocol and learn how to connect to IRC servers without using a special client. Once you have mastered this, you should find it a trivial task to write programs that connect to IRC.



You can find all of the Internet RFC (Request for Comments) documents on <http://www.faqs.org/rfcs>. You can search the archives by word or document number. RFC 1459 can be found at <http://www.faqs.org/rfcs/rfc1459.html>.

Fortunately, you do not need to know about the full specification to connect to an IRC server. Connecting to an IRC server requires only a few commands be sent. A good way of understanding how these commands work is to connect directly to an IRC server with Telnet and type these commands in directly. Telnet allows you to establish a TCP connection to a port on a remote machine and simply start typing in commands for the service listening on that port.

Most IRC servers run on port 6667, although you may find a few that operate on different port numbers to assist users who are stuck behind corporate firewalls. For this example, you can try connecting to the freenode IRC network by running Telnet from a command prompt with the following command-line parameters:

```
% telnet irc.freenode.net 6667
```

If the connection was successful, you will see the server respond with something similar to this:

```
NOTICE AUTH :*** Looking up your hostname...
NOTICE AUTH :*** Found your hostname, welcome back
NOTICE AUTH :*** Checking ident
NOTICE AUTH :*** No identd (auth) response
```

Although the socket is physically connected to the IRC server, you still need to do a couple of things. The IRC server will need to know your login, your real name, and the nickname you want to use.

The NICK command is used to set your nickname. This is pretty straightforward, so if I wanted my nickname to be “Paul,” I would type the following into the Telnet window and press Enter:

```
NICK Paul
```

If the nickname you choose is already in use on the server, you will be told so and you will have to keep sending the same command with a different nickname each time, until you find one that is available. This is the kind of message you will see if the nickname is already in use:

```
:kornbluth.freenode.net 433 * Paul :Nickname is already in use.
```

The `USER` command is used to set the login, user modes, and real name. If I wanted to have a login of “paul,” I would type the following, followed by Enter:

```
USER paul 8 * :Paul Mutton
```

Most servers these days respect commands from the updated IRC RFC 2812. This `USER` command makes use of some features specified in this updated document. In particular, the 8 is a numeric mode parameter that is used to automatically set user modes when registering with the server. This parameter is a bit mask, with bit 2 representing user mode `w` and bit 3 representing user mode `i`, so using a value of 8 means that you are asking the server to set you to be invisible. Currently, only these two bits have any significance. Also note that the text after the `:` is where you would enter your real name.

After successfully sending the `NICK` and `USER` commands, the server will send several lines of text to you. If nothing seems to be happening for a while, don't worry—the server may impose an artificial delay of up to a minute if it did not find an Ident server running on your machine. At first, the lines sent from the server may look rather confusing, but you may recognize some of them as being part of the message of the day. You are now connected to the IRC server!



Ident (Identification Protocol) is documented in RFC 1413. See <http://www.faqs.org/rfcs/rfc1413.html> for more details.

Now that you are connected, you are able to perform anything a fully functional IRC client would be capable of doing.

Staying Alive

Sometimes it is difficult for IRC servers to keep track of who is still connected. One trick they employ is to send `PING` commands to clients that have not exhibited any recent activity. The client is expected to respond with a `PONG` message to effectively say, “Hey, I'm still here!” If the client does not respond in a timely fashion, the server will close the connection. As a general rule, the `PONG` reply must include the arguments that were sent as part of

the PING command from the server. So if you were to receive the following message:

```
PING :kornbluth.freenode.net
```

You would reply with the following PONG command:

```
PONG :kornbluth.freenode.net
```

Joining Channels and Sending Messages

The usual IRC client commands like /join and /msg won't work here, as you are dealing with the raw protocol. You can think of your Telnet connection as a form of primitive IRC client—you can still do anything an IRC client can do, but it just looks a little ugly and the commands are different. Despite this, it is still pretty easy to join a channel and send messages to other users. To join the channel #irchacks, you just have to enter:

```
JOIN #irchacks
```

If all went well, you should see the IRC server replying with a few lines of text. IRC clients use these lines to determine who is in the channel so they can update their user lists. For now though, you needn't worry about how to parse that information.

Sending messages is slightly less intuitive and is achieved with the PRIVMSG command. As you might expect from the name, this command can be used to send private messages to other users; however, it is also the command used to send messages to entire channels. To send a message to the channel #irchacks, you could try the following:

```
PRIVMSG #irchacks :Hello everybody!
```

Sending private messages is just as easy—simply use the recipient's nickname in place of the channel name. If you want to send a private message to the user with the nickname “Dave,” you would enter:

```
PRIVMSG Dave :Hi Dave.
```

When you get bored with showing off your newfound protocol skills, you can quit from the server with the QUIT command. Using this command will cause the server to close your connection.

```
QUIT
```

The QUIT command can also take an optional parameter. The parameter must be immediately preceded by a : character. Whatever you supply here will be displayed to other users as your reason for quitting from the server:

```
QUIT :Telnet sucks!
```

Whichever method you use to quit from the server, before it disconnects you, it will respond with something along the lines of:

```
ERROR :Closing Link: Paul (Client Quit)
```

It is also possible for you to close the connection by closing Telnet, but this does not allow you to specify a reason for quitting.

While all commands should be capitalized, you may find that most servers are quite relaxed about this policy. Now that you know what to type to connect to IRC, the rest of the hacks in this chapter will show you how to create programs that connect to an IRC server.