

IRC HACKS™

100 Industrial-Strength Tips & Tools



O'REILLY®

Paul Mutton

HACK
#32

A Simple Perl IRC Client

Build a simple IRC robot that connects to an IRC server and joins a channel of your choosing.

If you have managed to connect directly to an IRC server with Telnet, you are probably ready to use this knowledge and start writing your own programs that connect to IRC automatically. In this example, you will build a simple IRC bot that connects to an IRC server and joins a channel. The term “*bot*” is commonly used to describe an automated IRC client and is a contraction of “*robot*.” You will get the bot to output information as it receives it from the server.

Many people find that Perl is a very suitable language for making simple IRC bots, as it is easy to use regular expressions to parse the data from the server. A single TCP socket is used to send and receive the text-based data, so you will need to use `IO::Socket`.

This will be quite a simple bot. All it has to do is connect to the server and join a channel. For this implementation, you must be aware that the IRC RFC states that each command or message must end with a return and new line (i.e., `\r\n`). To avoid getting disconnected by the server, you will also need to make sure that it responds appropriately to PING messages.

The Code

Enter the following code into your favorite plain text editor and save it as *irc.pl* or the like:

```
#!/usr/local/bin/perl -w
# irc.pl
# A simple IRC robot.
# Usage: perl irc.pl

use strict;

# We will use a raw socket to connect to the IRC server.
use IO::Socket;

# The server to connect to and our details.
my $server = "irc.freenode.net";
my $nick = "simple_bot";
my $login = "simple_bot";

# The channel which the bot will join.
my $channel = "#irchacks";

# Connect to the IRC server.
my $sock = new IO::Socket::INET(PeerAddr => $server,
```

```

PeerPort => 6667,
Proto => 'tcp') or
    die "Can't connect\n";

# Log on to the server.
print $sock "NICK $nick\r\n";
print $sock "USER $login 8 * :Perl IRC Hacks Robot\r\n";

# Read lines from the server until it tells us we have connected.
while (my $input = <$sock>) {
    # Check the numerical responses from the server.
    if ($input =~ /004/) {
        # We are now logged in.
        last;
    }
    elsif ($input =~ /433/) {
        die "Nickname is already in use.";
    }
}

# Join the channel.
print $sock "JOIN $channel\r\n";

# Keep reading lines from the server.
while (my $input = <$sock>) {
    chop $input;
    if ($input =~ /^PING(.*)$/i) {
        # We must respond to PINGs to avoid being disconnected.
        print $sock "PONG $1\r\n";
    }
    else {
        # Print the raw line received by the bot.
        print "$input\n";
    }
}

```

Running the Hack

Simply invoke the script on the command line with no arguments at all, like so:

```
% perl irc.pl
```

The Results

After the script has connected successfully, you will see it outputting all of the lines it receives from the IRC server, for example:

```

1 :simple_bot!identd@82-69-0-0.dsl.in-addr.zen.co.uk JOIN :#irchacks
2 :calvino.freenode.net 332 simple_bot #irchacks :IRC Hacks channel.
3 :calvino.freenode.net 333 simple_bot #irchacks Jibbler 1075562584
4 :calvino.freenode.net 353 simple_bot @ #irchacks :simple_bot DeadEd golbeck

```

A Simple Perl IRC Client

```
Jibbler dg Monty Wilmer
5 :calvino.freenode.net 366 simple_bot #irchacks :End of /NAMES list.
6 :Jibbler!~pjm2@torax.ukc.ac.uk PRIVMSG #irchacks :Hi simple_bot
7 :Monty!~monty@myrtle.ukc.ac.uk PRIVMSG #irchacks :It's simple_bot!
```

This may look quite confusing at first, but you can look at the lines one by one and work out what it means. The first line announces that the script has joined the channel `#irchacks`. Lines 2–5 contain information about this channel, such as what the channel topic is, who set it and when it was set, and who is in the channel. Lines 6 and 7 show messages being sent by other users in the channel.

You may like to use Perl’s regular expression features to process the output into a neater format, as you may not want to display all of this information. Taking a closer look at line 6 again:

```
:Jibbler!~pjm2@torax.ukc.ac.uk PRIVMSG #irchacks :Hi simple_bot
```

You can see that the message came from a user named Jibbler, with the login `pjm2` and connecting from `torax.ukc.ac.uk`. The `PRIVMSG` indicates that this is a normal message being sent to the channel `#irchacks`. The contents of the message are “Hi simple_bot”.