

Building IRC Bots



+



=



History

Must I feed you with every detail?

Check it yourself on Wikipedia!

What we need

- Registered IRC channel (for public server)
- Web server running PHP (can be a laptop too)
- IRC client (mIRC or WeeChat for CLI junkies)
- Keyboard

How to connect

- We will use irc server: irc.rizon.net, port 6667
- fsockopen() - works best for windows and linux
(find it in <http://www.php.net/quickref.php>)

```
<?php
```

```
// Opening the socket to the Rizon network
```

```
$socket = fsockopen("irc.rizon.net", 6667);
```

```
?>
```

Establishing connection with IRC

- After setting the socket variable, we need to send our auth info using fputs()

```
<?php
```

```
$socket = fsockopen("irc.rizon.net", 6667);
```

```
// send auth headers
```

```
fputs($socket, "USER SEC-Cbot sec-c.orgt SEC-C :SEC-C bot\n");
```

```
fputs($socket, "NICK CM-bot\n");
```

```
?>
```

- This will open the connection and tell the irc server who the bot is

See RFC details: http://www.irchelp.org/irchelp/rfc/chapter4.html#c4_1_3

Establishing connection with IRC

- Now we need to specify to which channel the bot will connect to

```
<?php
```

```
...
```

```
fputs($socket,"JOIN #sec-c-bot\n");
```

```
?>
```

Note: I registered the channel above for this example, you may have to do the same.

Keeping an idle connection

Problems:

- Script will die when it finished executing
- 30 sec max execution time in PHP

Solutions:

- Use an endless loop :)
- `set_time_limit()` for extending execution time

Keeping an idle connection

```
<?php
```

```
// Prevent php from stopping the script after 30 sec  
set_time_limit(0);
```

```
// open socket and send connection info
```

```
$socket = fsockopen("irc.rizon.net", 6667);
```

```
fputs($socket, "USER SEC-Cbot sec-c.org SEC-C :SEC-C bot\n");
```

```
fputs($socket, "NICK SECC-bot\n");
```

```
fputs($socket, "JOIN #sec-c-bot\n");
```

```
// make endless while loop
```

```
while(1) {
```

```
    // rest of the script goes here
```

```
}
```

```
?>
```

Reading data from server

- Now, to understand what the server is sending we will use `fgets()` to store the responses in a variable.
- `fgets()` needs the socket and number of bytes it can read

```
<?php
```

```
...
```

```
while($data = fgets($socket, 128)) {  
    echo $data;  
}  
?>
```

Reading data from server

- We can enhance this by using `nl2br()` to convert new-line chars to `
`
- Adding `flush()` to dump output buffer, so w/e was processed already will be displayed.

```
<?php
```

```
...
```

```
while($data = fgets($socket, 128)) {
```

```
    echo nl2br($data);
```

```
    flush();
```

```
}
```

```
?>
```

PING - PONG

- IRC uses a technique called PING/PONG to check your connection
- We need to recognize the PING and send back a PONG!
- We use `explode()` for blowing the received data (string) to individual words separated by " " (white space)
- Server will send:
PING :Unique-Code
- We need to send:
PONG: Unique-Code

PING - PONG

```
<?php
...
while($data = fgets($socket, 128)) {
    echo nl2br($data);
    flush();

    $array = explode(" ", $data);

    if ($array[0] == "PING") {
        fputs($socket, "PONG " . $array[1] . "\n");
    }
}
?>
```

Understanding commands

- The structure of IRC output:
:Nickname!Host@name PRIVMSG #sec-c-bot :The message
- Also, each line is followed by a `\n` and `\r` (new line and carriage return) so we need to clean that

```
<?php
```

```
...
```

```
$command = trim ($array[3]);
```

```
if ($command == "!:speak") {
```

```
    fputs ($socket, "PRIVMSG " . $array[2] . " :This is SEC-C!");
```

```
}
```

```
?>
```

The finished bot

```
<?php
set_time_limit(0);
$socket = fsockopen("irc.rizon.net", 6667);
fputs($socket, "USER SEC-Cbot sec-c.org SEC-C :SEC-C bot\n");
fputs($socket, "NICK SECC-bot\n");
fputs($socket, "JOIN #sec-c-bot\n");
while (1) {

while($data = fgets($socket, 128)) {
    echo nl2br($data);
    flush();
    $array = explode(" ", $data);
    if ($array[0] == "PING\n") {
        fputs($socket, "PONG " . $array[1] . "\n");
    }
    $command = trim ($array[3]);
    if ($command == "!:speak") {
        fputs ($socket, "PRIVMSG " . $array[2] . " :This is SEC-C!\n");
    }
}
}
?>
```

FIN

- Try adding your own commands now.
- You can also link it to other web based services using APIs
- The Bot can also be programmed to execute local system commands, see if you can figure out how.
- Presentation and source code available on www.SEC-C.org