How to make a webserver with netcat (nc)

The netcat tool `nc` can operate as a TCP client. Because HTTP works over TCP, `nc` can be used as an HTTP server! Because `nc` is a UNIX tool, we can use it to make custom web servers: servers which return any HTTP headers you want, servers which return the response very slowly, servers which return invalid HTTP, etc. You can also use `nc` as a quick-and-dirty static file server.

Here's an example. Run your web server by telling `nc` to listen for new connections on port 8000:

```
$ nc -l 8000
```

Then run your web browser. Here I use `curl` but you could also use Chrome etc:

```
$ curl localhost:8000/index.html
```

Back at `nc`, you'll see the HTTP request come through from `curl`:

```
$ nc -l 8000
GET /index.html HTTP/1.1
Host: localhost:8000
User-Agent: curl/7.54.0
Accept: */*
```

`nc` is now waiting for you to type the response! Type out the following:

```
HTTP/1.1 200 Everything Is Just Fine
Server: netcat!
Content-Type: text/html; charset=UTF-8

<!doctype html>
<html>
<body>
<h1>A webpage served with netcat</h1>
</body>
</html>
```

Once you start typing the HTML, you'll see it come line-by-line in your `curl` command. When you've finished typing the HTTP response, hit `Ctrl-D`. This tells `nc` to close the TCP connection and exit. The server is no more!

To run a persistent static server without typing anything in, write your HTTP response to a file like `index.http`:

```
$ cat index.http
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Server: netcat!

<!doctype html>
<html><body><h1>A webpage served by netcat</h1></body></html>
```

Then run `nc` in an infinite loop to serve this file for every response:

```
$ while true; do cat index.http | nc -l 8000; done
```

As an example of a "weird web server" you can make with `nc`, you can simulate a very slow web server. Use `pv --rate-limit 10` to read the file at 10 bytes per second:

```
while true; do pv --rate-limit 10 index.http | nc -l 8000; done
```

If you view this in Chrome, you can see Chrome's "progressive rendering"!

Coding Math

More by Jim

- Your syntax highlighter is wrong
- Granddad died today
- The Three Ts of Time, Thought and Typing: measuring cost on the web
- I hate telephones
- The sorry state of OpenSSL usability
- The dots do matter: how to scam a Gmail user
- My parents are Flat-Earthers
- How Hacker News stays interesting
- Project C-43: the lost origins of asymmetric crypto
- The hacker hype cycle
- The inception bar: a new phishing method
- Time is running out to catch COVID-19
- A probabilistic pub quiz for nerds
- Smear phishing: a new Android vulnerability

Tagged #programming, #unix, #networking. Found an error? Edit this page.