Onsite: Embedded Systems - Pair Programming (HTTP server)

-1

Created by Benjamin Bercovitz
Last updated Apr 02, 2020 by Dennis Kwok • Analytics

Problem

Build a basic HTTP server in C.

Time: 90 minutes. If it runs over or there is not enough time scheduled then it is ok to submit by email afterwards.

Pre-interview Setup

Share this section of information with the interviewee. The easiest way is to paste this section and the next into email (you can ask them for their email in the zoom chat).

You will need to use your own computer for this exercise. You will be asked to share your screen so prepare things so that we don't see anything you don't want us to.

Environment setup:

You will need a working C compiler and GNU make or similar. You probably have it installed, but if not, the ubuntu package `build-essential` provides these on ubuntu linux.

Note that all of this works on MacOSX or whatever other POSIX thing you are using.

starter Makefile:

```
# build an executable named httpserver for httpserver.c
CC=gcc
CFLAGS=-Wall -g
OBJFILES=httpserver.o
TARGET=httpserver

$(TARGET): $(OBJFILES)
	$(CC) $(CFLAGS) -o $(TARGET) $(OBJFILES) -lm
clean:
	rm -f $(OBJFILES) $(TARGET)
```

Also, create the blank C file `httpserver.c`

Please be ready to share your screen.

The goal of this project is to create a rudimentary HTTP/1.0 server that listens on port 5000.

This is the desired output from the following command:

```
(echo 'GET /sum?a=1&b=12 HTTP/1.0'; echo) | nc localhost 5000
```

```
HTTP/1.0 200 OK

13
```

Requirements:

1. Responds to simple GET requests with the URL `/sum?a=1&b=12` in a loop
2. Each time returns an HTTP response that has the result. In this case it would be the string `13` (the sum of the two numbers)
3. Does not have unsafely used buffers

What is HTTP?

Request:

```
HTTP-METHOD PATH PROTOCOL
[optional headers, separated by newline]
blank line to denote end-of-headers
```

```
GET / HTTP/1.0
```

Response:

```
PROTOCOL STATUS-CODE STATUS
[optional headers, separated by newline]
blank line to denote end-of-headers
[content]
```

```
HTTP/1.0 200 OK

Hello World!
```

What references are allowed:

- the manpages for example on http://man7.org/linux/man-pages/ (you can still use google with this filter: site:man7.org)
  - of particular interest:
  - http://man7.org/linux/man-pages/man2/socket.2.html
  - http://man7.org/linux/man-pages/man2/bind.2.html
  - http://man7.org/linux/man-pages/man2/listen.2.html
  - http://man7.org/linux/man-pages/man2/accept.2.html
  - http://man7.org/linux/man-pages/man7/ip.7.html
- cppreference.com (you can still use google with this filter: site:cppreference.com)
- Your interviewer

What references are specifically not allowed:

- stack overflow
- random websites

The basic outline of your program should be:

1. create a socket (you will want an AF_INET socket for address family: internet)
2. bind the socket to an address (you can use INADDR_LOOPBACK for localhost)
    a. make sure to htons()/htonl() to convert the address and port number to network byte order
3. put the socket in listen mode
4. accept new connections on that socket
5. do something when a connection is established

Tricky things:

`struct sockaddr` is a placeholder struct (kind of a C polymorphism pattern), see `man bind`. You will actually need to use the version of it that is called `struct sockaddr_in`, see `man ip`.

Read carefully what the third argument to `accept()` is.

Notes for interviewer

> Notes for interviewer

Rubric

> Rubric

Scoring:

> Scoring