

**《计算机综合训练》课程作业**

|  |  |  |
| --- | --- | --- |
| **学 号：** | **16281123** | |
| **姓 名：** | **周大业** | |
| **专 业：** | **计算机科学与技术** | |
| **学 院：** | **计算机与信息技术学院** | |
|  |  |  |
|  |  |  |
| **提交日期：** | **2018年07月20日** | |

### 产品需求

用C语言实现一个基于HTTP/1.0协议的简易HTTP Server, 并实现在对应的命令行操作，命令规范详见SWS BSD General Commands Manual[1]

### 程序文件清单

./docs

cgi.js

index.html

form.html

./src

./getarg

CMakeLists.txt

getarg.c

getarg.h

bunker.c

bunker.h

CMakeLists.txt

main.c

### API说明

使用Linux中socket编程的库，socket, bind, listen等函数频繁使用。

以及使用stat函数查看文件的状态

**bunker.h**

void error(const char \*s);

报出对应的错误

int strcasecmp(const char \*s1, const char \*s2);

忽略大小写的字符串比较，s1和s2完全匹配返回0

size\_t read\_line(int sock, char \*buffer, size\_t size);

按行读取socket传输进来的数据

int start\_server(const char \*address, unsigned short port, const char \*dir,

const char \*log\_file, bool debug);

根据对应的参数，对socket file descriptor进行bind, listen处理

void accept\_request(int client, const char \*dir);

解析socket传来的数据

void execute\_cgi(int client, unsigned short \*codep, const char \*path,

const char \*method, const char \*query\_string);

对于socket数据中可执行cgi的文件执行cgi

void send\_file(int client, unsigned short \*codep, const char \*path);

向客户端发送文件

void send\_headers(int client, const char \*path);

向客户端发送请求头

void cat(int client, const char \*path);

Linux中cat功能的实现

throw\_xxx\_xxx为抛出各种异常的函数。

void throw\_unimplement\_method(int client, unsigned short \*codep);

void throw\_internal\_server\_error(int client, unsigned short \*codep);

void throw\_not\_found(int client, unsigned short \*codep);

void throw\_forbidden(int client, unsigned short \*codep);

void throw\_bad\_request(int client, unsigned short \*codep);

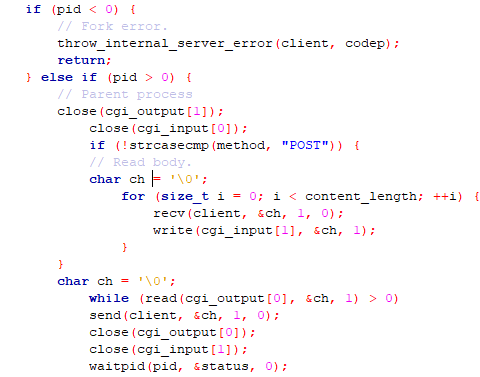
void stop\_server(int server);

### 设计难点

* 1. **CGI的处理以及与客户端的通信**

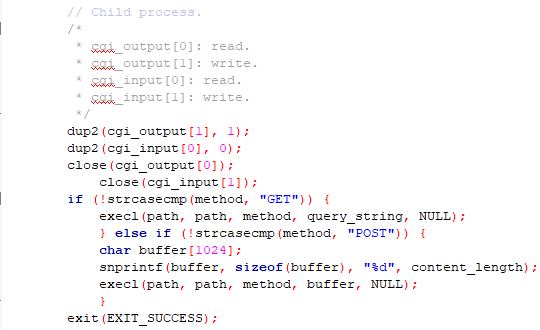
建立了两个管道，利用了管道的半双工特性，fork当前进程，子进程处理CGI文件，父进程通过管道与客户端进行通信。

**(1)父进程与客户端的通信**



通过pid判断是父进程还是子进程，如果当前method是POST，则读取body中的内容并等待子进程关闭或者信号发出。

**（2）子进程和客户端的通信**

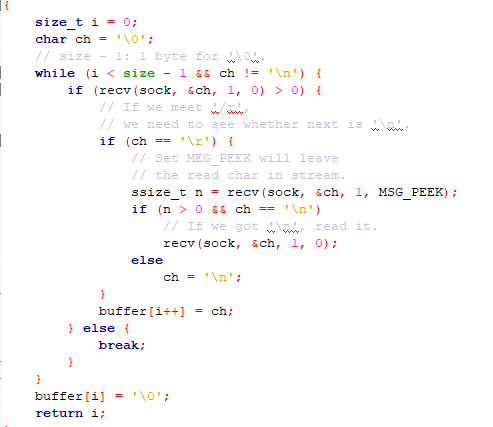


关闭cgi\_output管道读端和cgi\_input的写端，如果当前方法为GET方法，则将当前query\_string传给cgi文件执行。

如果当前方法为POST方法，则读取body，并退出进程。

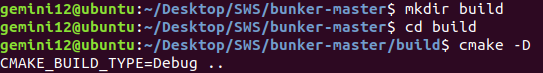
* 1. **Read\_line函数的设计**

当遇到\r, \r\n, 和\n时，读取结束，具体实现如下



### 程序执行流程以及程序测试

首先，创建文件并进入在命令行输入对对应参数





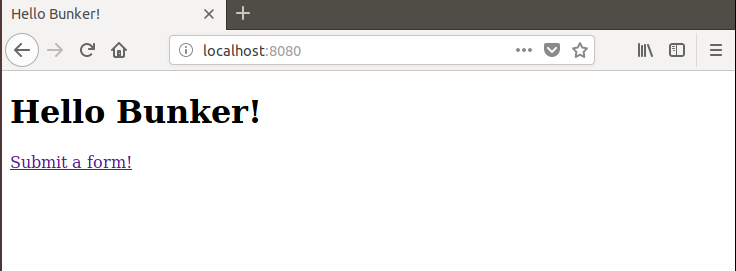


通过上述命令，通过cmake生成exe文件，在命令行下指定端口为8080， cgi文件的存放位置为 ../docs, log文件为当前目录下的./bunker.log

* 1. **浏览器输入测试**

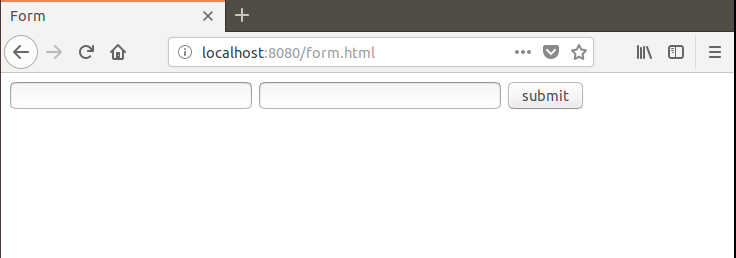
浏览器输入localhost:8080

进入如下界面

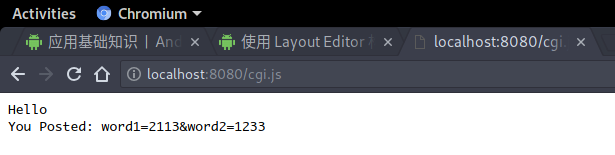


* 1. **POST方法测试**

form中输入数据



运行结果如下：

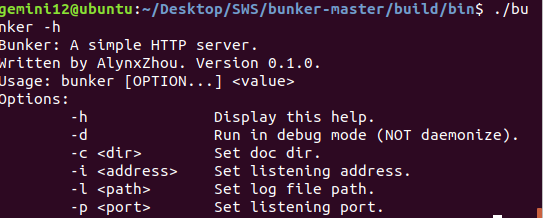


* 1. **GET方法测试**

****

**d. 命令行测试**

**-h命令测试：**



**-l命令测试：**

**在docs文件夹中的bunker.log文件记录了相关log**



### 系统要求和运行要求:

编程语言：C，Node.js（用于执行自定义 CGI 脚本）

开发环境：Arch Linux

运行环境：任意版本的Linux，需要gcc以及Node.js的相关环境，使用cmake做编译管理。

### 参考资料

Linux socket网络编程API

tinyhttpd