UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
CÂMPUS TRÊS LAGOAS - MS
SISTEMAS DE INFORMAÇÃO
REDES DE COMPUTADORES II

Nome do Aluno: Luiz Fernando Postingel Quirino		RGA: 2016.0743.003-7
Assunto: Atividade 03	Data(s): 25-04-2021	

Atividade 03 - Programação com Sockets C (TCP) - 14/04/2021

Implementação

Foi implementado um HTTP Server simples, que serve um arquivo *".html"*.

Para testes e utilização

Compilar com os parâmetros padrão

```
$ gcc -ansi -Wall -pedantic server.c -o server
```

Executando com as definições padrão:

```
$ ./server
```

Executando com as definições (arquivo e porta) customizados:

```
$ ./server arquivo.html 8001
```

Ambiente de desenvolvimento

- Arch Linux x86_64
- Kernel 5.11.16-arch1-1 SMP
- GCC 10.2.0

Dificuldades encontradas

Demorei para encontrar a documentação que queria em uma linguagem acessível aos meus conhecimentos, apesar de ter um acesso tranquilo ao inglês, as terminologias que se utilizava em alguns materiais, como o código que usei de base, disponível neste vídeo[1] do canal *Eduonix Learning Solutions*, por exemplo, me pareciam muito soltos, apesar de funcionarem nos testes.

Depois de alguma pesquisa, encontrei uma cópia gratuita e em PDF pesquisável do livro *Programando em C. Para linux Unix e Windows*[2], que no capítulo 23 explica de forma bem clara e objetiva as terminologias e funções que haviam me causado confusão na literatura/exemplos que havia encontrado, inclusive do código usei como guia para compreender compreender e desenvolver uma solução ao problema apresentado.

Há algumas ressalvas e melhorias possíveis, entre elas, o arquivo .html lido necessita ter apenas uma linha, dado que é feita a leitura com fgets, esta é uma problemática que poderia ter sido resolvida em uma implementação com mais tempo.

1 Anexo I - Código fonte utilizado

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>


int isNumeric(const char *str)
{
    while(*str != '\0')
    {
        if(*str < '0' || *str > '9')
            return 0;
        str++;
    }
    return 1;
}


int main(int argc, char const *argv[])
{

  FILE *html_data;
  char response_data[4096], input[255], filename[255];
  char http_header[4096] = "HTTP/1.1 200 OK\r\nContent-Type: text/html; charset=
   UTF-8\r\n\r\n";
  char http_content[8192] = "";
  int server_socket, client_socket, port;
  struct sockaddr_in server_address;

  /* Verifica parametros passados e permite customizar algumas
   * opcoes de acesso
   */
  if (argc < 3 )
  {
    port=8080;
    strcpy(filename, "index.html");

    printf("\n\tEntre com um nome para o arquivo html(sem a extensao)");
    printf("\n\tdeixe vazio para index.html\n\t");
    fgets (input, 255, stdin);
    input[strlen(input) - 1] = '\0';
    if(strlen(input))
    {
      strcpy(filename, input);
      strcat(filename, ".html");
    }

    printf("\n\tEntre com um numero para a porta");
    printf("\n\tdeixe em branco para 8080\n\t");
    fgets (input, 255, stdin);
    input[strlen(input) - 1] = '\0';
    if(strlen(input) && isNumeric(input))
    {
        port = atoi(input);
    }

```

```c
  } else {
    strcpy(filename, argv[1]);
    port = atoi(argv[2]);
  }


  /* cria o socket
   */
  server_socket = socket(AF_INET, SOCK_STREAM, 0);

  /* define o endereco do server
   */
  server_address.sin_family = AF_INET;
  server_address.sin_port = htons(port);
  server_address.sin_addr.s_addr = INADDR_ANY;

  bind(server_socket,(struct sockaddr *)&server_address, sizeof(server_address));
  listen(server_socket, 5);

  /* executa enquanto nao recebe um sinal de cancelamento
   */
  printf("\n\t http://localhost:%d", port);
  printf("\n\t Pressione Ctrl+C para finalizar\n");
  while (1)
  {
    /* Limpa e reseta o cabecalho
     */
    memset(http_content, '\0', sizeof(http_content));
    strcat(http_content, http_header);

    /* abre um arquivo para servir, e le seu conteudo
     * ao user o fgets, o html precisa estar formatado em apenas uma linha
       * com um pouco mais de tempo, teria sido interessante implementar a
   melhoria
       */
    html_data = fopen(filename, "r");
    fgets(response_data, 4096, html_data);
    fclose(html_data);

    strcat(http_content, response_data);

    client_socket = accept(server_socket, NULL, NULL);
    send(client_socket, http_content, sizeof(http_content), 0);
    close(client_socket);
  }

  return 0;
}
```

Observação: preferi não desmembrar o código e não criar bibliotecas em arquivos separados para facilitar os testes.

Referências

[1] Eduonix Learning Solutions. *Socket Programming Tutorials In C For Beginners | Part 2 | Eduonix*. URL: https://www.youtube.com/watch?v=mStnzIEprH8.

[2] Marcos Aurelio Pchek Laureano. *Programando em C. Para Linux, Unix e Windows*. URL: http://www.mlaureano.org/livro/Programando_C_conta.pdf.