

11711BCC008

Matheus José da Costa

Servidor de Arquivos

A partir de alguns conceitos estudados na disciplina de Sistemas Operacionais, para a etapa final, foi requisitado um Servidor de Arquivos como tema do meu Trabalho Final.

Este servidor possui duas operações básicas: o cliente buscar um arquivo do servidor e uma outra cuja finalidade é enviar um arquivo para o servidor. Estas operações, para o cliente, são conhecidas como GET e POST, e ele poderá fazer uma requisição a partir de uma interface a qual ele informará o arquivo e diretório de onde deseja buscar ou enviar um arquivo para o servidor. Após dada requisição, o cliente se comunicará com o servidor e, caso seja uma operação de busca de arquivo, então o servidor verificará do seu lado se dado arquivo existe no diretório informado. Caso seja uma operação de envio de arquivo, então será verificado se o arquivo existe do lado do cliente, e se positiva a mensagem, o cliente processará o envio do arquivo.

Para tanto, foram implementadas duas versões do programa Servidor, uma single-thread e uma outra, multi-thread. Na versão single-thread, o servidor apenas consegue atender a uma única requisição. Já na versão multi-thread, é aceito mais de uma conexão simultânea.

Além do mais, este programa foi escrito em Linguagem C utilizando Sockets -- para o envio de informações entre cliente e servidor -- e o protocolo TCP.

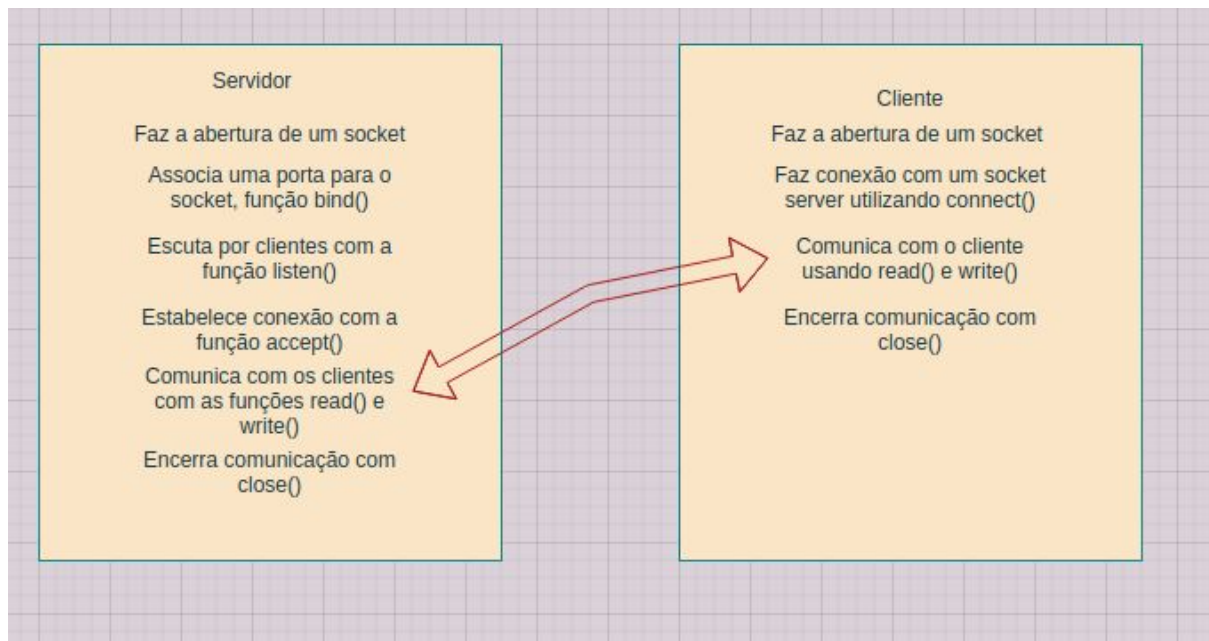
Sockets

Acerca do Sockets, temos que, hoje em dia, a maioria das comunicações via rede utiliza-se o modelo de cliente-servidor.

Para tanto, o cliente necessita saber o IP e porta em que o servidor se encontra. Além do mais, o servidor não sabe da existência do cliente. Então, sabendo-se disso, qualquer programa consegue se comunicar no servidor a partir do IP e porta.

A comunicação via Sockets funciona como se fosse uma ligação telefônica, a qual existe uma pessoa a qual será contactada -- servidor -- e uma a qual tenta fazer a ligação -- conexão com o servidor. Nessa analogia, é como se o nosso IP e porta fossem o número telefone discado. O cliente, sabendo dessas informações, se conecta ao servidor e conversar entre si. Quando o cliente não quer mais comunicar com o servidor, então ele desconecta.

Segue um exemplo mais visual feito no site ***draw.io***:



Os sockets operam a partir de dois protocolos: TCP e UDP. O protocolo TCP é usado quando necessita-se saber que uma informação de fato chegou ao destinatário. O UDP é usado quando um volume de dados é muito grande e não precisa-se ter garantia de recebimento de todos os dados.

Explicação do funcionamento do código:

Execução:

Para executar o código do servidor, basta o seguinte comando, no linux:

Servidor single-thread: **`gcc -o server server.c -pthread`**

Servidor multi-thread: **`gcc -o server server.c -pthread`**

Cliente: **`gcc -o client client.c`**

Estando com o servidor disponível para conexões, e conectando algum cliente nesse servidor, conseguimos fazer as seguintes operações, POST e GET:

`POST [diretório em que se encontra o arquivo] [nome do arquivo]`

`GET [diretório em que se encontra o arquivo] [nome do arquivo]`

Servidor:

Tanto para a versão single-thread do servidor, quanto a multi-thread, há aspectos em comum, como é o caso da criação de um socket.

Basicamente, temos, primeiramente, uma struct do servidor e uma struct cliente -- o qual se conectará ao servidor:

```
struct sockaddr_in servidor, cliente;
```

Então criamos um socket para o servidor e verificamos se tudo ocorreu como esperado:

```
_socket = socket(AF_INET, SOCK_STREAM, 0);  
if (_socket == -1) {  
    printf("[-] Não foi possível criar o _socket\n");  
    return -1;  
}
```

Após isso, atribuímos ao servidor o seu devido IP,, especificamos a família (ARPANET) e a porta a qual ele escutará por clientes.

```
servidor.sin_family = AF_INET;  
servidor.sin_addr.s_addr = INADDR_ANY;  
servidor.sin_port = htons(portaServidor);
```

Depois, é feito um bind do servidor:

```
if (bind(_socket, (struct sockaddr *) &servidor, sizeof (servidor)) <  
0) {  
    puts("[-] Erro ao fazer bind, tente outra porta\n");  
    return -1;  
}
```

E, então, para a versão single thread se escuta por um cliente, enquanto na multi-thread, por 20 clientes, como o a seguir:

```
listen(_socket, 1);
```

```
listen(_socket, 20);
```

Logo em seguida, o servidor fica escutando por conexões, e, após algum cliente conectado, é processado as devidas operações de busca e envio de arquivos:

```
while ((conexao = accept(_socket, (struct sockaddr *) &cliente,  
(socklen_t*) & c))) {  
    int tamanho;
```

```

char respostaServidor[50];
char *mensagemPostOuGet, *mensagemNomeDiretorio;
// lendo dados enviados pelo cliente
//mensagem 1 recebido nome do arquivo

cliente_ip = inet_ntoa(cliente.sin_addr);
cliente_port = ntohs(cliente.sin_port);
printf("[+] Cliente conectou: %s : [ %d ]\n", cliente_ip,
cliente_port);

if ((tamanho = read(conexao, mensagem_post_ou_get, MAX_MSG)) <
0) {
    perror("[-] Erro ao receber dados do cliente: ");
    return NULL;
}

mensagem_post_ou_get[tamanho] = '\0';
printf("[+] O cliente falou sobre o Método: %s, tamanho %d\n",
mensagem_post_ou_get, tamanho);

mensagemPostOuGet = mensagem_post_ou_get;
//mensagem 2 - enviando confirmação que arquivo existe do lado
do cliente
write(conexao, mensagemPostOuGet, strlen(mensagemPostOuGet));
printf("[+] O servidor falou sobre ter recebido a mensagem de
post ou get, para o cliente: %s\n", mensagemPostOuGet);

if(keyfromstring(mensagem_post_ou_get) == 1){
    if ((tamanho = read(conexao, respostaNomeDiretorio,
MAX_MSG)) < 0) {
        perror("[-] Erro ao receber dados do cliente: ");
        return NULL;
    }
    respostaNomeDiretorio[tamanho] = '\0';
    diretorioParaBuscarArquivo = opendir(respostaNomeDiretorio);
    if(diretorioParaBuscarArquivo == NULL ){fprintf(stderr,
"Diretório %s não existe.\n", respostaNomeDiretorio);return -1;}
    printf("\n[+] O cliente falou sobre nome do diretório:
%s\n", respostaNomeDiretorio);
    mensagemNomeDiretorio = respostaNomeDiretorio;
    write(conexao, mensagemNomeDiretorio,
strlen(mensagemNomeDiretorio));

```

```
        printf("\n[+] O servidor falou sobre que recebeu o nome do
diretório: %s\n", mensagemNomeDiretorio);
    }

    switch(keyfromstring(mensagem_post_ou_get)){
        case 1:

            if (conexao < 0) {
                perror("[-] Erro ao receber conexao\n");
                return -1;
            }

            novaConexao = (int) malloc(1);
            novaConexao = conexao;

            connection_handler(novaConexao);

            puts("[+] GET: Conexão estabelecida");

            if (novaConexao < 0) {
                perror("[-] Não foi possível estabelecer conexão com
o cliente.");
                return 1;
            }
            break;
        case 2:
            if (conexao < 0) {
                perror("[-] Erro ao receber conexao\n");
                return -1;
            }

            novaConexao = (int) malloc(1);
            novaConexao = conexao;

            connection_client(novaConexao);

            puts("[+] POST: Conexão estabelecida");

            if (novaConexao < 0) {
                perror("[-] Não foi possível estabelecer conexão com
o cliente.");
                return 1;
            }
    }
```

```

        break;
    default:
        printf("Waiting..."); break;
    }
}

```

Nesta etapa, então o servidor recebe uma informação do método ao qual o cliente deseja fazer operação: GET ou POST. Então o servidor responde para o cliente com o nome do método. E então o cliente recebe essa mensagem e a confirma se chegou realmente okay, e segue o fluxo. O mesmo é feito para o nome do diretório e nome do arquivo.

Caso o método que o cliente mande seja GET, a função ***connection_handle*** será chamada:

```

void *connection_handler(void *_socket) {
    char *mensagem, *mensagemArquivoExiste;
    char respostaNomeArquivo[MAX_MSG];
    int tamanho;
    char respostaArquivoOkay[MAX_MSG];

    // lendo dados enviados pelo cliente
    //mensagem 1 recebido nome do arquivo
    if ((tamanho = read(conexao, respostaNomeArquivo, MAX_MSG)) < 0)
    {
        perror("Erro ao receber dados do cliente: ");
        return NULL;
    }

    respostaNomeArquivo[tamanho] = '\0';
    printf("\nO cliente falou sobre o arquivo: %s\n",
respostaNomeArquivo);

    mensagem = respostaNomeArquivo;
    write(conexao, mensagem, strlen(mensagem));
    printf("O servidor falou sobre ter recebido o nome do arquivo
para o cliente: %s\n", mensagem);

    char nomeArquivoAuxiliar[MAX_MSG];

    strncpy(nomeArquivoAuxiliar, respostaNomeArquivo, MAX_MSG);

    if(tamanho = read(conexao, respostaArquivoOkay, MAX_MSG) < 0){
        printf("Falha ao receber resposta\n");
        return -1;
    }
}

```

```

printf("A resposta é: %s", respostaArquivoOkay);

if (diretorioParaBuscarArquivo != NULL) {

    while ((arquivoProcurado =
readdir(diretorioParaBuscarArquivo)) != NULL) {

        stat(arquivoProcurado->d_name, &mystat);

        printf("Arquivo lido: %s, Arquivo procurado: %s\n",
arquivoProcurado->d_name, respostaNomeArquivo);
        if (strcmp(arquivoProcurado->d_name,
respostaNomeArquivo) == 0) {
            printf("Chegou aqui.");
            closedir(diretorioParaBuscarArquivo);

            arquivoProcurado = NULL;
            diretorioParaBuscarArquivo = NULL;
            diretorioParaBuscarArquivo =
opendir(respostaNomeDiretorio);

            mensagem = "200";
            //mensagem 2 - enviando confirmação que arquivo
existe

            write(conexao, mensagem, strlen(mensagem));
            printf("Servidor falou pro cliente que o arquivo
existe: %s", mensagem);

            //mensagem 3 - recebendo que arquivo OK do cliente
            read(conexao, mensagemArquivoExiste, MAX_MSG);
            printf("Servidor falou pro cliente que o arquivo
existe: %s", mensagemArquivoExiste);

            char arquivo[1024];
            strncpy(arquivo, respostaNomeDiretorio, 1024);
            strcat(arquivo, nomeArquivoAuxiliar);

            FILE * file = fopen(arquivo, "rb");
            if((fseek(file, 0, SEEK_END))<0){ printf("ERRO
DURANTE fseek"); }

            uint32_t len = (int) ftell(file);

```

```

        uint32_t converted_number = htonl(len);

        printf("[+] Tamanho do arquivo: %d",
converted_number);

        write(_socket, &converted_number,
sizeof(converted_number));

        int fd = open(arquivo, O_RDONLY);
        off_t offset = 0;
        int bytesEnviados = 0;

        if (fd == -1) {
            fprintf(stderr, "Erro ao abrir arquivo: %s",
strerror(errno));

            exit(EXIT_FAILURE);
        }

        while (((bytesEnviados = sendfile(conexao, fd,
&offset, BUFSIZ)) > 0) && (len > 0)) {

            fprintf(stdout, "[+] Servidor enviou %d bytes do
arquivo, offset agora é: %d e os dados restantes = %d\n",
bytesEnviados, (int)offset, len);
            len -= bytesEnviados;
            if (len <= 0) {
                fprintf(stdout, "[+] Servidor enviou %d
bytes do arquivo, offset agora : %d e os dados restantes = %d\n",
bytesEnviados, (int)offset, len);
                break;
            }
        }

        while (1) {

        }

    }
}if(arquivoProcurado==NULL) {
    mensagem = "404";

    printf("\n//*****//\n");
}

```



```

        printf("Arquivo \"%s\" não existe no diretório:
        \"%s\"\n", nomeArquivoAuxiliar, respostaNomeDiretorio);

        write(conexao, mensagem, strlen(mensagem));

        diretorioParaBuscarArquivo =
        opendir(respostaNomeDiretorio);

        while (1) {

        }

        close(conexao);

    }

    if (diretorioParaBuscarArquivo != NULL) {
        closedir(diretorioParaBuscarArquivo);
        diretorioParaBuscarArquivo = NULL;
    }
}

if (strcmp(respostaNomeArquivo, "bye\n") == 0) {
    close(conexao);
    printf("[+] Servidor finalizado...\n");
    return NULL;
}
}

```

Essa função, basicamente, recebe o nome do arquivo a partir do cliente, verifica se o arquivo existe localmente, no diretório especificado, e então pega-se o tamanho do arquivo em bytes e envia os bytes desse arquivo para o cliente a partir da função `sendfile`. Então recebe os bytes do arquivo e grava localmente. Caso não se encontre o arquivo, o servidor volta a escutar por novas operações. Após enviar todos os bytes do arquivo, o servidor volta a escutar por conexões.

Caso o cliente requisiite o envio de um arquivo, então o servidor cria uma função chamada `connection_client`, o qual receberá as devidas informações do cliente e armazenará, localmente, o arquivo.

```

void *connection_client(void *conexao) {
    char respostaServidor[MAX_MSG];
    int tamanho;
    char *mensagemEnviaNomeArquivoRequeridoParaServidor;
    char *mensagem;
    if ((tamanho = read(conexao, respostaServidor, MAX_MSG)) < 0) {

```

```

        perror("[-] Erro ao receber dados do cliente.");
        return NULL;
    }

    printf("[+] Servidor recebeu o nome do arquivo a ser gravado:
%s.\n", respostaServidor);

    write(conexao, respostaServidor, strlen(respostaServidor));
    printf("[+] O servidor falou sobre ter recebido a mensagem do
nome do arquivo: %s\n", respostaServidor);

    FILE *arquivoRecebido;
    arquivoRecebido = fopen(respostaServidor, "w");
    ssize_t len;
    char buffer[BUFSIZ];
    int quantidadeDeBytesRestanteParaSerGravado;

    memset(arquivoRecebido, 0, sizeof arquivoRecebido);
    memset(respostaServidor, 0, sizeof respostaServidor);

    //Recebendo respostaNomeArquivo do servidor
    //mensagemEnviaNomeArquivoRequeridoParaServidor 2 recebendo que
arquivo existe
    if((tamanho = read(conexao, respostaServidor, MAX_MSG)) < 0) {
        printf("[-] Falha ao receber respostaNomeArquivo\n");
        return -1;
    }

    printf("[+] Status: %s da existência do arquivo do lado do
cliente.\n", respostaServidor);
    if (strcmp(respostaServidor, "200") == 0) {

        mensagemEnviaNomeArquivoRequeridoParaServidor = "OK";
        //mensagemEnviaNomeArquivoRequeridoParaServidor 3 enviado ok
        write(conexao,
mensagemEnviaNomeArquivoRequeridoParaServidor,
strlen(mensagemEnviaNomeArquivoRequeridoParaServidor));
        //mensagemEnviaNomeArquivoRequeridoParaServidor 4 recebendo
o tamanho do arquivo;
        memset(respostaServidor, 0, sizeof respostaServidor);

        uint32_t received_int;
        read(conexao, &received_int, sizeof(received_int));

```

```

        quantidadeDeBytesRestanteParaSerGravado =
ntohl(received_int);
    }else{
        fprintf(stderr, "[-] Arquivo não encontrado no cliente.\n");
    }

    while (((len = recv(conexao, buffer, BUFSIZ, 0)) > 0) &&
(quantidadeDeBytesRestanteParaSerGravado > 0)) {
        fwrite(buffer, sizeof (char), len, arquivoRecebido);
        quantidadeDeBytesRestanteParaSerGravado -= len;
        fprintf(stdout, "[+] Recebidos %d bytes e aguardando: %d
bytes\n", len, quantidadeDeBytesRestanteParaSerGravado);
        if (quantidadeDeBytesRestanteParaSerGravado <= 0) {
            break;
        }
    }
    fclose(arquivoRecebido);
    close(conexao);
    if (strcmp(respostaServidor, "200") == 0){
        printf("[+] Servidor recebeu arquivo %s com sucesso!\n",
arquivoRecebido);
    }
}

```

Cliente:

Assim como o servidor, o cliente também cria um socket:

```

_socket = socket(AF_INET, SOCK_STREAM, 0);

if (_socket == -1) {
    printf("Nao foi possivel criar _socket\n");
    return -1;
}

```

```

char* portaServidorAuxiliar = argv[2];
int portaServidor = atoi(portaServidorAuxiliar);

servidor.sin_addr.s_addr = inet_addr(argv[1]);
servidor.sin_family = AF_INET;
servidor.sin_port = htons(portaServidor);

```

Atribui-se ao cliente o devido IP ao qual foi informado inicialmente pelo usuário ao executar, também a porta a qual quer se conectar e a família ARPANET.

Então o cliente faz uma conexão no servidor:

```
if (conexao = connect(_socket, (struct sockaddr *) &servidor, sizeof
(servidor)) < 0) {
    printf("Nao foi possivel conectar\n");
    return -1;
}
```

Após conectado no servidor, o cliente envia a sua operação (POST ou GET), requisitando um arquivo ou enviando um arquivo:

Então são enviadas as informações para o servidor, do método POST ou GET, nome do diretório e o nome do arquivo. Então o servidor vai confirmando para o cliente se chegou tudo okay. Após isso, o cliente então começa as operações, caso seja um GET, o cliente recebe o tamanho do arquivo para ter essa variável como controladora e saber quando parar de escrever no arquivo. E a cada byte enviado pelo servidor, o cliente escreve no arquivo em questão.

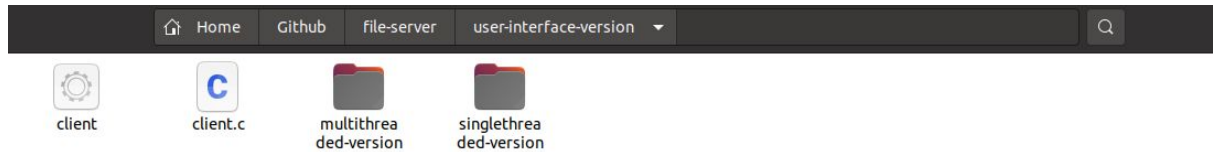
Caso seja um método POST, então o cliente verifica se o arquivo existe localmente e manda o tamanho para o servidor. Além de enviar os bytes do arquivo para o servidor gravar.

O servidor na prática

Versão single-thread:

Busca de um arquivo:

Pasta na qual o cliente se encontra antes de fazer quaisquer requisição para o servidor, seja de busca ou envio de arquivo:



Diretório /home/matheus/Documents mostrando que o arquivo ao qual o cliente requisitará, existe:



Servidor inicializa criando um socket na porta 4001:

```
matheus @ zup-2744 in ~/Github/file-server/user-interface-version/singlethreaded-version on git:multithread x [20:32:01] C:130
$ ./server 4001
[+] Bind efetuado com sucesso
[+] Aguardando por conexões...
```

Cliente conecta no servidor na porta 4001:

```
matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [20:32:41]
$ ./client 127.0.0.1 4001
Conectado no servidor
Seja bem vindo ao servidor de arquivos!
Operações disponíveis para efetuar no servidor: POST/GET
Use: [METODO POST OU GET] [DIRETORIO] [ARQUIVO]:

input$
```

Servidor após receber conexão:

```
matheus @ zup-2744 in ~/Github/file-server/user-interface-version/singlethreaded-version on git:multithread x [20:32:01] C:130
$ ./server 4001
[+] Bind efetuado com sucesso

[+] Aguardando por conexões...
[+] Cliente conectou: 127.0.0.1 : [ 59640 ]
```

Cliente manda uma requisição de uma busca de arquivo a partir do método GET, passando o nome do arquivo (exemplo-servidor-arquivos.pdf) e o diretório no qual se encontra, a partir do seguinte comando: **GET /home/matheus/Documents/ singlethread-example-get.pdf**

```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [20:55:17]
$ ./client 127.0.0.1 4001
Conectado no servidor
Seja bem vindo ao servidor de arquivos!
Operações disponíveis para efetuar no servidor: POST/GET
Use: [METODO POST OU GET] [DIRETORIO] [ARQUIVO]:

input$ GET /home/matheus/Documents/ singlethread-example-get.pdf
Cliente envia GET ou POST: GET
Resposta recebida do servidor sobre o método get ou post: GET, comparacao: 0
Envia nome do diretório: /home/matheus/Documents/
Resposta recebida do servidor sobre o nome do diretório: /home/matheus/Documents/
Cliente envia nome do arquivo: singlethread-example-get.pdf
Resposta recebida do servidor sobre o nome do arquivo: singlethread-example-get.pdf
Enviando que o arquivo está okay: OKResposta recebida da existencia do arquivo no servidor: status 200, 0
Aqui Zoka: OK[+] Recebidos 8192 bytes e aguardamos :- 32659 bytes
[+] Recebidos 8192 bytes e aguardamos :- 24467 bytes
[+] Recebidos 8192 bytes e aguardamos :- 16275 bytes
[+] Recebidos 8192 bytes e aguardamos :- 8083 bytes
[+] Recebidos 8083 bytes e aguardamos :- 0 bytes
Cliente finalizado com sucesso!
```

O servidor então processa a dada requisição do cliente, enviando pacotes do arquivo requisitado para ele:

```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version/singlethreaded-version on git:multithread x [20:53:03]
$ ./server 4001
[+] Bind efetuado com sucesso

[+] Aguardando por conexões...
[+] Cliente conectou: 127.0.0.1 : [ 54724 ]
[+] O cliente falou sobre o Método: GET, tamanho 3
[+] O servidor falou sobre ter recebido a mensagem de post ou get, para o cliente: GET

[+] O cliente falou sobre nome do diretório: /home/matheus/Documents/

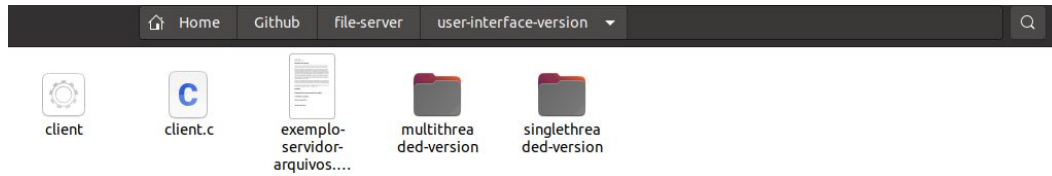
[+] O servidor falou sobre que recebeu o nome do diretório: /home/matheus/Documents/
[+] GET: Conexão estabelecida

O cliente falou sobre o arquivo: singlethread-example-get.pdf
O servidor falou sobre ter recebido o nome do arquivo para o cliente: singlethread-example-get.pdf
A resposta é: OKArquivo lido: programacao-orientada-objetos-2, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: message.pdf, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: Pratica 07.odt, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: Horários Disciplinas Remotas.xlsx, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: feedback_poo2.pdf, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: Untitled.mdj, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: mds.mdj, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: message2.pdf, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: bug-engagement, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: dump_qa_20200902.sql, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: auto_conhecimento, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: lab05-mds.mfj, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: account_fee_story, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: Deploy Extensions QA.postman.collection.json, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: singlethread-example-post.pdf, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: extensions, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: ..., Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: collections, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: terapia, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: Untitled 1.pdf, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: DocuSign_TRACE_-_MATHEUS_JOSE_DA_COSTA.pdf_P.pdf, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: ., Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: materias-feitas-ufu, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: jenkins-project, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: gestao-empresarial, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: acadeny, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: E_mail_de_Zup_IT_Innovation_Projeto_Auto_Conhecimento_Fase_1.pdf, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: dump_qa_2.sql, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: html-docs, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: trabalho-mds-diagrama-sequencia.mfj, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: variaveis_ambiente.zip, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: regex, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: Fragment.mfj, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: engenharia-software, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: Emissão da Carteira de Identidade 2ª e outras vias Estado de Minas.pdf, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: 09-Automated Testing In DevOps-Ing-P17-2016.pdf, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: scripts, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: programacao-internet, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: packets.pcapng, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: security, Arquivo procurado: singlethread-example-get.pdf
Arquivo lido: singlethread-example-get.pdf, Arquivo procurado: singlethread-example-get.pdf
Chegou aqui.Servidor falou pro cliente que o arquivo existe: 200Servidor falou pro cliente que o arquivo existe: (null)[+] Tamanho do arquivo: -1818296320
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 16384 e os dados restantes = 32659
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 24576 e os dados restantes = 24467
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 32768 e os dados restantes = 16275
[+] Servidor enviou 8083 bytes do arquivo, offset agora é: 40851 e os dados restantes = 8083
[+] Servidor enviou 8083 bytes do arquivo, offset agora : 40851 e os dados restantes = 0

zsh
```

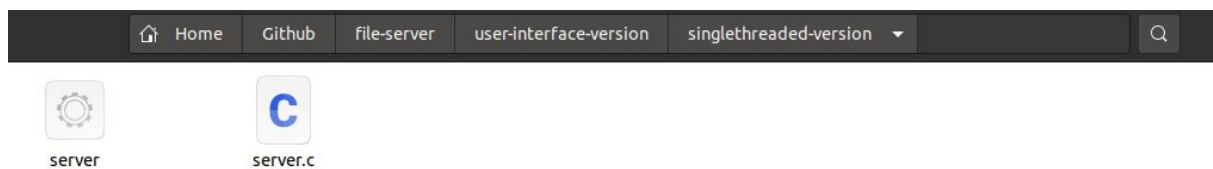
Servidor termina de processar e enviar os pacotes para o cliente.

Após finalizar, então é possível observar, na pasta raiz do cliente, que o arquivo foi transferido com sucesso:

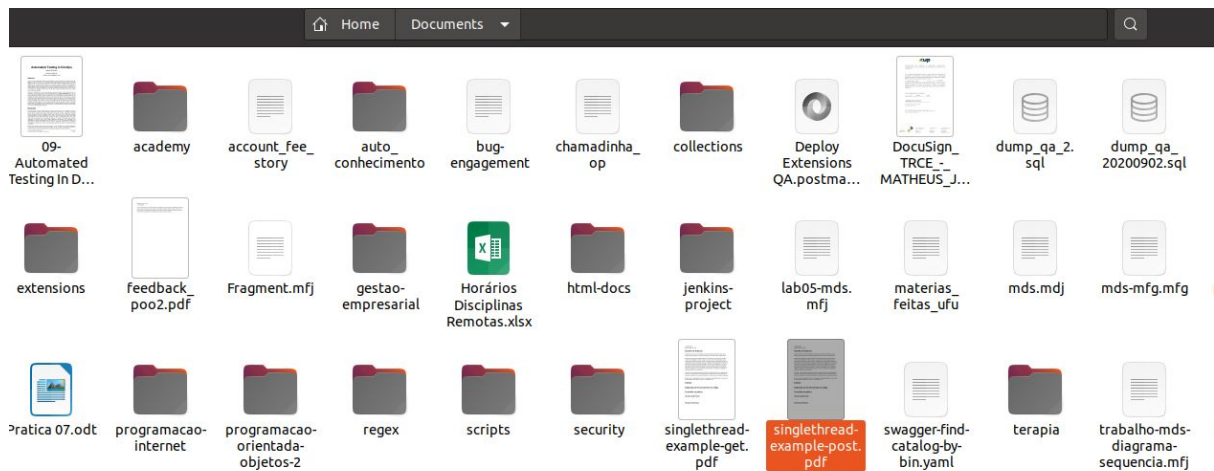


Envio de um arquivo para o servidor:

Diretório no qual o servidor single thread se encontra antes da requisição:



Diretório /home/matheus/Documents mostrando que o arquivo ao qual o cliente requisitará, existe:



Servidor iniciado na porta 4001:

```
matheus @ zup-2744 in ~/Github/file-server/user-interface-version/singlethreaded-version on git:multithread x [20:32:01] C:130
$ ./server 4001
[+] Bind efetuado com sucesso
[+] Aguardando por conexões...
```

Cliente conecta no servidor na porta 4001 e requisita o envio de um arquivo para o servidor a partir do comando: **POST /home/matheus/Documents/ singlethread-example-post.pdf**

```
matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [20:56:20]
$ ./client 127.0.0.1 4001
Conectado no servidor
Seja bem vindo ao servidor de arquivos!
Operações disponíveis para efetuar no servidor: POST/GET
Use: [METODO POST OU GET] [DIRETORIO] [ARQUIVO]:
input$ POST /home/matheus/Documents/ singlethread-example-post.pdf
```

Cliente verifica se o arquivo existe na sua máquina no diretório especificado e então manda para o servidor:

```

# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread * [20:56:20]
$ ./client 127.0.0.1 4001
Conectado no servidor
Seja bem vindo ao servidor de arquivos!
Operações disponíveis para efetuar no servidor: POST/GET
Use: [METODO POST OU GET] [DIRETORIO] [ARQUIVO]:

input$ POST /home/matheus/Documents/ singlethread-example-post.pdf
Dados enviados POST
Resposta recebida do servidor sobre o método get ou post: POST, comparacao: 0
O cliente enviou nome do arquivo singlethread-example-post.pdf para o servidor
Resposta recebida do servidor sobre o nome do arquivo: singlethread-example-post.pdf
Arquivo lido: programacao-orientada-objetos-2, Arquivo procurado: singlethread-example-post.pdf, Comparação: -3
Arquivo lido: message.pdf, Arquivo procurado: singlethread-example-post.pdf, Comparação: -6
Arquivo lido: Pratica 07.odt, Arquivo procurado: singlethread-example-post.pdf, Comparação: -35
Arquivo lido: Horários Disciplinas Remotas.xlsx, Arquivo procurado: singlethread-example-post.pdf, Comparação: -43
Arquivo lido: feedback_poo2.pdf, Arquivo procurado: singlethread-example-post.pdf, Comparação: -13
Arquivo lido: Untitled.mdj, Arquivo procurado: singlethread-example-post.pdf, Comparação: -30
Arquivo lido: mds.mdj, Arquivo procurado: singlethread-example-post.pdf, Comparação: -6
Arquivo lido: message2.pdf, Arquivo procurado: singlethread-example-post.pdf, Comparação: -6
Arquivo lido: bug-engagement, Arquivo procurado: singlethread-example-post.pdf, Comparação: -17
Arquivo lido: dump_qa_20200902.sql, Arquivo procurado: singlethread-example-post.pdf, Comparação: -15
Arquivo lido: auto_conhecimento, Arquivo procurado: singlethread-example-post.pdf, Comparação: -18
Arquivo lido: lab05-mds.mfj, Arquivo procurado: singlethread-example-post.pdf, Comparação: -7
Arquivo lido: account_fee_story, Arquivo procurado: singlethread-example-post.pdf, Comparação: -18
Arquivo lido: Deploy Extensions QA.postman_collection.json, Arquivo procurado: singlethread-example-post.pdf, Comparação: -47
Arquivo lido: singlethread-example-post.pdf, Arquivo procurado: singlethread-example-post.pdf, Comparação: 0
Arquivo existe lado do clientemensagem 200[+] Tamanho do arquivo: -1818296320[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 8192 e os dados restantes = 32659
[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 16384 e os dados restantes = 24467
[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 24576 e os dados restantes = 16275
[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 32768 e os dados restantes = 8083
[+] Cliente enviou 8083 bytes do arquivo, offset agora é: 40851 e os dados restantes = 0
[+] Cliente enviou 8083 bytes do arquivo, offset agora é: 40851 e os dados restantes = 0
[+] Cliente enviou todos os dados do arquivo com sucesso.
[+] Cliente terminando...

```

Servidor recebe os dados do arquivo do cliente:

```

# matheus @ zup-2744 in ~/Github/file-server/user-interface-version/singlethreaded-version on git:multithread * [20:58:22] C:130
$ ./server 4001
[+] Bind efetuado com sucesso

[+] Aguardando por conexões...
[+] Cliente conectou: 127.0.0.1 : [ 49108 ]
[+] O cliente falou sobre o Método: POST, tamanho 4
[+] O servidor falou sobre ter recebido a mensagem de post ou get, para o cliente: POST
[+] Servidor recebeu o nome do arquivo a ser gravado: singlethread-example-post.pdf.
[+] O servidor falou sobre ter recebido a mensagem do nome do arquivo: singlethread-example-post.pdf
[+] Status: 200 da existência do arquivo do lado do cliente.
[+] Recebidos 8192 bytes e aguardando: 32659 bytes
[+] Recebidos 8192 bytes e aguardando: 24467 bytes
[+] Recebidos 8192 bytes e aguardando: 16275 bytes
[+] Recebidos 8192 bytes e aguardando: 8083 bytes
[+] Recebidos 8083 bytes e aguardando: 0 bytes
[+] POST: Conexão estabelecida

zsh

```

Pasta na qual o servidor single thread se encontra:



Versão multi-thread:

Busca de arquivo:

Servidor inicializa criando um socket na porta 4001:

```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version/multithreaded-version on git:multithread x [21:04:16]
$ ./server 4002
[+] Bind efetuado com sucesso
```

Dois clientes se conectam no servidor na porta 4001:

```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [21:04:23]
$ ls
client client.c multithreaded-version singlethreaded-version

# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [21:04:23]
$ ./client 127.0.0.1 4002
Conectado no servidor
Seja bem vindo ao servidor de arquivos!
Operações disponíveis para efetuar no servidor: POST/GET
Use: [METODO POST OU GET] [DIRETORIO] [ARQUIVO]:

input$

# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [21:04:41]
$ ./client 127.0.0.1 4002
Conectado no servidor
Seja bem vindo ao servidor de arquivos!
Operações disponíveis para efetuar no servidor: POST/GET
Use: [METODO POST OU GET] [DIRETORIO] [ARQUIVO]:

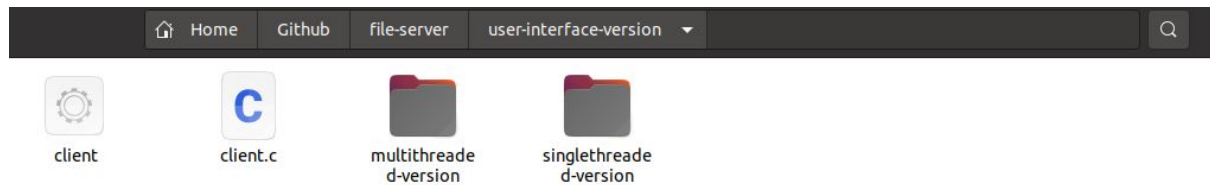
input$ █
```

Servidor após receber duas conexões:

```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version/multithreaded-version on git:multithread x [21:04:16]
$ ./server 4002
[+] Bind efetuado com sucesso

[+] Aguardando por conexões...
[+] Cliente conectou: 127.0.0.1 : [ 33538 ]
[+] Cliente conectou: 127.0.0.1 : [ 34184 ]
```

Diretório no qual se encontra o cliente antes de fazer quaisquer operações de busca ou envio de arquivo:



Cientes requisitam dois arquivos do servidor a partir dos comandos:

GET /home/matheus/Documents/ multithreaded-example-get1.pdf

GET /home/matheus/Documents/ multithreaded-example-get2.pdf

```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [21:07:08]
$ ./client 127.0.0.1 4080
Conectado no servidor
Seja bem vindo ao servidor de arquivos!
Operações disponíveis para efetuar no servidor: POST/GET
Use: [METODO POST OU GET] [DIRETORIO] [ARQUIVO]:

input$ GET /home/matheus/Documents/ multithreaded-example-get1.pdf
Cliente envia GET ou POST: GET
Resposta recebida do servidor sobre o método get ou post: GET, comparacao: 0
Envia nome do diretório: /home/matheus/Documents/
Resposta recebida do servidor sobre o nome do diretório: /home/matheus/Documents/
Cliente envia nome do arquivo: multithreaded-example-get1.pdf
Resposta recebida do servidor sobre o nome do arquivo: multithreaded-example-get1.pdf
Enviando que o arquivo está okay: OKResposta recebida da existencia do arquivo no servidor: status 200, 0
Aqui Zoka: OK[+] Recebidos 8192 bytes e aguardamos :- 32659 bytes
[+] Recebidos 8192 bytes e aguardamos :- 24467 bytes
[+] Recebidos 8192 bytes e aguardamos :- 16275 bytes
[+] Recebidos 8192 bytes e aguardamos :- 8083 bytes
[+] Recebidos 8083 bytes e aguardamos :- 0 bytes
Cliente finalizado com sucesso!

# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [21:09:56]
$

zsh
```

```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [21:08:48]
$ ./client 127.0.0.1 4080
Conectado no servidor
Seja bem vindo ao servidor de arquivos!
Operações disponíveis para efetuar no servidor: POST/GET
Use: [METODO POST OU GET] [DIRETORIO] [ARQUIVO]:

input$ GET /home/matheus/Documents/ multithreaded-example-get2.pdf
Cliente envia GET ou POST: GET
Resposta recebida do servidor sobre o método get ou post: GET, comparacao: 0
Envia nome do diretório: /home/matheus/Documents/
Resposta recebida do servidor sobre o nome do diretório: /home/matheus/Documents/
Cliente envia nome do arquivo: multithreaded-example-get2.pdf
Resposta recebida do servidor sobre o nome do arquivo: multithreaded-example-get2.pdf
Enviando que o arquivo está okay: OKResposta recebida da existencia do arquivo no servidor: status 200, 0
Aqui Zoka: OK[+] Recebidos 8192 bytes e aguardamos :- 32659 bytes
[+] Recebidos 8192 bytes e aguardamos :- 24467 bytes
[+] Recebidos 8192 bytes e aguardamos :- 16275 bytes
[+] Recebidos 8192 bytes e aguardamos :- 8083 bytes
[+] Recebidos 8083 bytes e aguardamos :- 0 bytes
Cliente finalizado com sucesso!

# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [21:09:56]
$
```

Servidor processa os arquivos enviados e envia para o cliente:

Processamento do primeiro cliente:

```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version/multithreaded-version on git:multithread x [21:09:22] C:\130
$ ./server 4002
[+] Bind efetuado com sucesso

[+] Aguardando por conexões...
[+] Cliente conectou: 127.0.0.1 : [ 56728 ]
[+] O cliente falou sobre o Método: GET, tamanho 3
[+] O servidor falou sobre ter recebido a mensagem de post ou get, para o cliente: GET

[+] O cliente falou sobre nome do diretório: /home/matheus/Documents/

[+] O servidor falou sobre que recebeu o nome do diretório: /home/matheus/Documents/

O cliente enviou para o servidor o arquivo: multithreaded-example-get1.pdf, multithreaded-example-get1.pdf
O servidor falou sobre ter recebido o nome do arquivo para o cliente: multithreaded-example-get1.pdf
A resposta é: OK/[Arquivo lido: programacao-orientada-objetos-2, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: message.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: Pratica 07.odt, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: Horários Disciplinas Remotas.xlsx, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: feedback_poo2.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: Untitled.mdj, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: mds.mdj, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: message2.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: bug-engagement, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: dump_qa_20200902.sql, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: auto_conhecimento, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: lab05-mds.mfj, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: account_fee_story, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: Deploy Extensions QA.postman_collection.json, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: singlethread-example-post.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: extensions, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: ..., Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: collections, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: terapia, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: Untitled 1.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: DocuSign_TRCE_-_MATHEUS_JOSE_DA_COSTA.pdf_P.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: ., Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: materias-feitas-ufu, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: jenkins-project, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: gestao-empresarial, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: academy, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: multithreaded-example-get2.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: E_mail_de_Zup_IT_Innovation_Projeto_Auto_Conhecimento_Fase_1.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: dump_qa_2.sql, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: html-docs, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: trabalho-mds-diagrama-sequencia.mfj, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: variaveis_ambiente.zip, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: regex, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: Fragment.mfj, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: engenharia-software, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: Emissão da Carteira de Identidade 2ª e outras vias Estado de Minas.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: multithreaded-example-get1.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Servidor falou pro cliente que o arquivo existe: 200
Servidor recebeu do cliente confirmação do arquivo: OK
[+] Tamanho do arquivo: -1818296320[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 8192 e os dados restantes = 40851
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 16384 e os dados restantes = 32659
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 24576 e os dados restantes = 24467
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 32768 e os dados restantes = 16275
[+] Servidor enviou 8083 bytes do arquivo, offset agora é: 40851 e os dados restantes = 8083
[+] Servidor enviou 8083 bytes do arquivo, offset agora : 40851 e os dados restantes = 0
[+] Cliente conectou: 127.0.0.1 : [ 36368 ]
[+] O cliente falou sobre o Método: GET, tamanho 3
[+] O servidor falou sobre ter recebido a mensagem de post ou get, para o cliente: GET
zsh
```

Processamento do segundo cliente:

```
Arquivo lido: multithreaded-example-get2.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: E_mail_de_Zup_IT_Innovation_Projeto_Auto_Conhecimento_Fase_1.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: dump_qa_2.sql, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: html-docs, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: trabalho-mds-diagrama-sequencia.mfj, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: variaveis_ambiente.zip, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: regex, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: Fragment.mfj, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: engenharia-software, Arquivo procurado: multithreaded-example-get1.pdf
Arquivo lido: Emissão_da_Carteira_de_Identidade_2ª_e_outras_vias_Estado_de_Minas.pdf, Arquivo procurado: multithreaded-example-get1.p
Arquivo lido: multithreaded-example-get1.pdf, Arquivo procurado: multithreaded-example-get1.pdf
Servidor falou pro cliente que o arquivo existe: 200
Servidor recebeu do cliente confirmação do arquivo: OK
[+] Tamanho do arquivo: -1818296320[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 8192 e os dados restantes = 40851
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 16384 e os dados restantes = 32659
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 24576 e os dados restantes = 24467
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 32768 e os dados restantes = 16275
[+] Servidor enviou 8083 bytes do arquivo, offset agora é: 40851 e os dados restantes = 8083
[+] Servidor enviou 8083 bytes do arquivo, offset agora : 40851 e os dados restantes = 0
[+] Cliente conectou: 127.0.0.1 : [ 36368 ]
[+] O cliente falou sobre o Método: GET, tamanho 3
[+] O servidor falou sobre ter recebido a mensagem de post ou get, para o cliente: GET

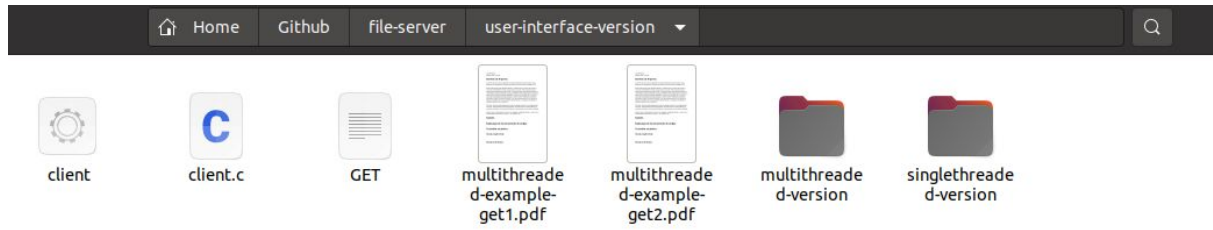
[+] O cliente falou sobre nome do diretório: /home/matheus/Documents/

[+] O servidor falou sobre que recebeu o nome do diretório: /home/matheus/Documents/

O cliente enviou para o servidor o arquivo: multithreaded-example-get2.pdf, multithreaded-example-get2.pdf
O servidor falou sobre ter recebido o nome do arquivo para o cliente: multithreaded-example-get2.pdf
A resposta é: OK[Arquivo lido: programacao-orientada-objetos-2, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: message.pdf, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: Pratica 07.odt, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: Horários Disciplinas Remotas.xlsx, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: feedback_poo2.pdf, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: Untitled.mdj, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: mds.mdj, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: message2.pdf, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: bug-engagement, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: dump_qa_20200902.sql, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: auto_conhecimento, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: lab05-mds.mfj, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: account_fee_story, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: Deploy Extensions QA.postman_collection.json, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: singlethread-example-post.pdf, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: extensions, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: .., Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: collections, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: terapia, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: Untitled 1.pdf, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: DocuSign_TRCE_-_MATHEUS_JOSE_DA_COSTA.pdf.P.pdf, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: .., Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: materias_feitas_ufr, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: jenkins-project, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: gestao-empresarial, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: academy, Arquivo procurado: multithreaded-example-get2.pdf
Arquivo lido: multithreaded-example-get2.pdf, Arquivo procurado: multithreaded-example-get2.pdf
Servidor falou pro cliente que o arquivo existe: 200
Servidor recebeu do cliente confirmação do arquivo: OK
[+] Tamanho do arquivo: -1818296320[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 8192 e os dados restantes = 40851
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 16384 e os dados restantes = 32659
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 24576 e os dados restantes = 24467
[+] Servidor enviou 8192 bytes do arquivo, offset agora é: 32768 e os dados restantes = 16275
[+] Servidor enviou 8083 bytes do arquivo, offset agora é: 40851 e os dados restantes = 8083
[+] Servidor enviou 8083 bytes do arquivo, offset agora : 40851 e os dados restantes = 0

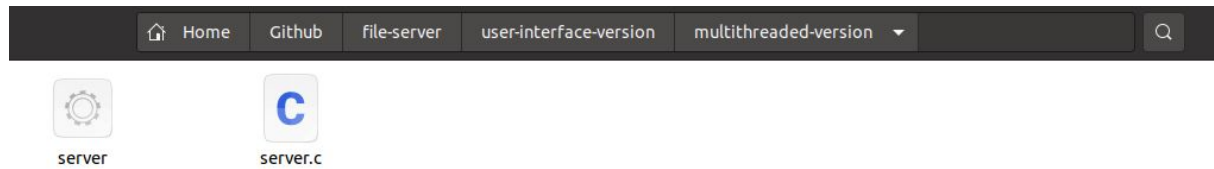
zsh
```

Diretório o qual se encontra o cliente após a requisição dos arquivos ***multithreaded-example-get1.pdf*** e ***multithreaded-example-get2.pdf***.

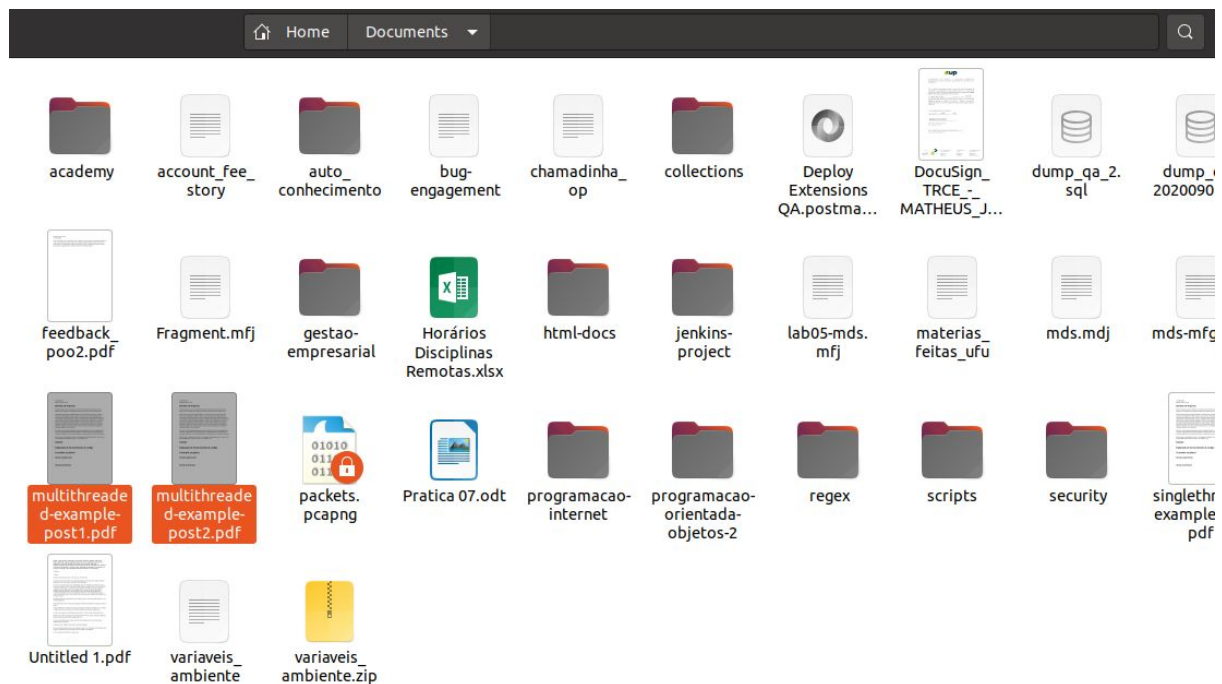


Envio de um arquivo para o servidor:

Diretório o qual se encontra o servidor multi-thread antes de quaisquer operações:



Diretório /home/matheus/Documents o qual se encontra os dois arquivos (***multithreaded-example-post1.pdf*** e ***multithreaded-example-post2.pdf***) que serão enviados para o servidor:



Servidor após receber duas conexões:

```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version/multithreaded-version on git:multithread x [21:11:57] C:130
$ ./server 4002
[+] Bind efetuado com sucesso

[+] Aguardando por conexões...
[+] Cliente conectou: 127.0.0.1 : [ 55738 ]
[+] Cliente conectou: 127.0.0.1 : [ 40242 ]
```

O primeiro cliente verifica se os arquivos existem no diretório **/home/matheus/Documents/** e envia para o servidor:

```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [21:12:02]
$ ./client 127.0.0.1 4002
Conectado no servidor
Seja bem vindo ao servidor de arquivos!
Operações disponíveis para efetuar no servidor: POST/GET
Use: [METODO POST OU GET] [DIRETORIO] [ARQUIVO]:

input$ POST /home/matheus/Documents/ multithreaded-example-post1.pdf
Dados enviados POST
Resposta recebida do servidor sobre o método get ou post: POST, comparacao: 0
O cliente enviou nome do arquivo multithreaded-example-post1.pdf para o servidor
Resposta recebida do servidor sobre o nome do arquivo: multithreaded-example-post1.pdf
Arquivo lido: programacao-orientada-objetos-2, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: 3
Arquivo lido: message.pdf, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: -16
Arquivo lido: Pratica 07.odt, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: -29
Arquivo lido: Horários Disciplinas Remotas.xlsx, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: -37
Arquivo lido: feedback_poo2.pdf, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: -7
Arquivo lido: Untitled.mdj, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: -24
Arquivo lido: mds.mdj, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: -17
Arquivo lido: message2.pdf, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: -16
Arquivo lido: bug-engagement, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: -11
Arquivo lido: dump_qa_20200902.sql, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: -9
Arquivo lido: auto_conhecimento, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: -12
Arquivo lido: lab05-mds.mfj, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: -1
Arquivo lido: multithreaded-example-post1.pdf, Arquivo procurado: multithreaded-example-post1.pdf, Comparação: 0
Arquivo existe lado do clientemensagem 200[+] Tamanho do arquivo: -1818296320[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 8192 e os dados restantes = 32659
[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 16384 e os dados restantes = 24467
[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 24576 e os dados restantes = 16275
[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 32768 e os dados restantes = 8083
[+] Cliente enviou 8083 bytes do arquivo, offset agora é: 40851 e os dados restantes = 0
[+] Cliente enviou 8083 bytes do arquivo, offset agora é: 40851 e os dados restantes = 0
[+] Cliente enviou todos os dados do arquivo com sucesso.
[+] Cliente terminando...

# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [21:13:18]
$ 
zsh
```

O segundo cliente verifica se o arquivo existe localmente e envia para o servidor:

```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [21:12:04]
$ ./client 127.0.0.1 4002
Conectado no servidor
Seja bem vindo ao servidor de arquivos!
Operações disponíveis para efetuar no servidor: POST/GET
Use: [Método POST OU GET] [Diretório] [Arquivo]:

Input$ POST /home/matheus/Documents/ multithreaded-example-post2.pdf
Dados enviados POST
Resposta recebida do servidor sobre o método get ou post: POST, comparação: 0
O cliente enviou nome do arquivo multithreaded-example-post2.pdf para o servidor
Resposta recebida do servidor sobre o nome do arquivo: multithreaded-example-post2.pdf
Arquivo lido: programacao-orientada-objetos-2, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: 3
Arquivo lido: message.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -16
Arquivo lido: Pratica 07.odt, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -29
Arquivo lido: Horários Disciplinas Remotas.xlsx, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -37
Arquivo lido: feedback poo2.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -7
Arquivo lido: Untitled.ndj, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -24
Arquivo lido: mds.mdj, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -17
Arquivo lido: message2.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -16
Arquivo lido: bug-engagement, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -11
Arquivo lido: dump_ga_20200902.sql, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -9
Arquivo lido: auto_conhecimento, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -12
Arquivo lido: lab05-mds.nfj, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -1
Arquivo lido: multithreaded-example-post1.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -1
Arquivo lido: account_fee_story, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -12
Arquivo lido: Deploy Extensions QA.postman_collection.json, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -41
Arquivo lido: singlethread-example-post.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: 6
Arquivo lido: extensions, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -8
Arquivo lido: ., Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -63
Arquivo lido: collections, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -10
Arquivo lido: terapia, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: 7
Arquivo lido: Untitled 1.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -24
Arquivo lido: DocuSign_TRCE_-_MATHEUS_JOSE_DA_COSTA.pdf_P.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -41
Arquivo lido: ., Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -63
Arquivo lido: materias-feitas-ufu, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -20
Arquivo lido: jenkins-project, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -3
Arquivo lido: gestao-empresarial, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -6
Arquivo lido: academy, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -12
Arquivo lido: multithreaded-example-get2.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -9
Arquivo lido: E_mail_de_Zup_IT_Innovation_Projeto_Auto_Conhecimento_Fase_1.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -40
Arquivo lido: dump_ga_2.sql, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -9
Arquivo lido: html-docs, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -5
Arquivo lido: trabalho-nds-diagrama-sequencia.nfj, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: 7
Arquivo lido: variavets-ambiente.zip, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: 9
Arquivo lido: regex, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: 5
Arquivo lido: Fragment.nfj, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -39
Arquivo lido: engenharia-software, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -8
Arquivo lido: Emissão_da_Carteira_de_Identidade_2ª_e_outras_vias_Estado_de_Minas.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -40
Arquivo lido: multithreaded-example-get1.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -9
Arquivo lido: 09-Automated Testing In DevOps-Ing-P17-2016.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: -61
Arquivo lido: scripts, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: 6
Arquivo lido: programacao-internet, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: 3
Arquivo lido: multithreaded-example-post2.pdf, Arquivo procurado: multithreaded-example-post2.pdf, Comparação: 0
Arquivo existe lado do clientemensagem 200[+] Tamanho do arquivo: -1818296320[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 8192 e os dados restantes = 32659
[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 16384 e os dados restantes = 24467
[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 24576 e os dados restantes = 16275
[+] Cliente enviou 8192 bytes do arquivo, offset agora é: 32768 e os dados restantes = 8083
[+] Cliente enviou 8083 bytes do arquivo, offset agora é: 40851 e os dados restantes = 0
[+] Cliente enviou 8083 bytes do arquivo, offset agora é: 40851 e os dados restantes = 0
[+] Cliente enviou todos os dados do arquivo com sucesso.
[+] Cliente terminando...


# matheus @ zup-2744 in ~/Github/file-server/user-interface-version on git:multithread x [21:14:19]
$
zsh
```


Servidor recebe os arquivos:


```
# matheus @ zup-2744 in ~/Github/file-server/user-interface-version/multithreaded-version on git:multithread x [21:11:57] C:130
$ ./server 4002
[+] Bind efetuado com sucesso


[+] Aguardando por conexões...
[+] Cliente conectou: 127.0.0.1 : [ 55738 ]
[+] Cliente conectou: 127.0.0.1 : [ 40242 ]
[+] O cliente falou sobre o Método: POST, tamanho 4
[+] O servidor falou sobre ter recebido a mensagem de post ou get, para o cliente: POST
[+] Servidor recebeu o nome do arquivo a ser gravado: multithreaded-example-post2.pdf.
[+] O servidor falou sobre ter recebido a mensagem do nome do arquivo: multithreaded-example-post2.pdf
[+] Status: 200 da existência do arquivo do lado do cliente.
[+] Recebidos 8192 bytes e aguardando: 32659 bytes
[+] Recebidos 8192 bytes e aguardando: 24467 bytes
[+] Recebidos 8192 bytes e aguardando: 16275 bytes
[+] Recebidos 8192 bytes e aguardando: 8083 bytes
[+] Recebidos 8083 bytes e aguardando: 0 bytes
[+] POST: Conexão estabelecida
[+] O cliente falou sobre o Método: POST, tamanho 4
[+] O servidor falou sobre ter recebido a mensagem de post ou get, para o cliente: POST
[+] Servidor recebeu o nome do arquivo a ser gravado: multithreaded-example-post1.pdf.
[+] O servidor falou sobre ter recebido a mensagem do nome do arquivo: multithreaded-example-post1.pdf
[+] Status: 200 da existência do arquivo do lado do cliente.
[+] Recebidos 8192 bytes e aguardando: 32659 bytes
[+] Recebidos 8192 bytes e aguardando: 24467 bytes
[+] Recebidos 8192 bytes e aguardando: 16275 bytes
[+] Recebidos 8192 bytes e aguardando: 8083 bytes
[+] Recebidos 8083 bytes e aguardando: 0 bytes
[+] POST: Conexão estabelecida
```


Diretório o qual se encontra o servidor após o envio dos arquivos ***multithreaded-example-post1.pdf*** e ***multithreaded-example-post2.pdf*** pelo cliente.


 Home


 Github

 file-server


 user-interface-version


 multithreaded-version ▾




multithreaded-example-post1.pdf


multithreaded-example-post2.pdf


server


server.c