UNIVERSIDADE FEDERAL DE MINAS GERAIS

REDES DE COMPUTADOR -TRABALHO PRÁTICO 1

GABRIEL SOUZA DE OLIVEIRA

1. INTRODUÇÃO

O advento da tecnologia Smart Grid tem transformado radicalmente a indústria de energia elétrica, possibilitando a integração de fontes renováveis, otimização da distribuição e melhoria na eficiência energética. Este relatório documenta o desenvolvimento de um sistema cliente-servidor que simula as operações vitais de uma Smart Grid, permitindo o controle, monitoramento e interação com sensores em tempo real.

2. ARQUITETURA DO SERVIDOR (MTU)

O sistema é baseado em uma arquitetura servidor-cliente, utilizando a comunicação via TCP/IP. O servidor (MTU) é responsável pelo processamento das solicitações recebidas do cliente (RTU) e pela gestão dos sensores. O cliente, por sua vez, interage com o usuário e envia comandos para o servidor.

- **Limitações do servidor**

1- O servidor não recebe comandos do teclado.

```
gabriel89067@DESKTOP-7OHVILA:~/programas$ ./server v4 90900
bound to IPv4 0.0.0.0 25364, waiting connections
teste
oi
```

2- O limite de clientes na fila é de 15.

```
////////////// listen() /////////// fica aguardando ate 15 clientes na fila
if (0 != listen(s, 15))
{
    logexit("listen");
}
```

3- O servidor só suporta armazenar até 100 sensores.

```
struct equipamento equipamentos[100];
```

- **Funcionalidades do servidor**

Após o recebimento da mensagem do “client” via comando recv() , como mostrado abaixo:

```
char buf[BUFSZ];
memset(buf, 0, BUFSZ);
size_t count = recv(csock, buf, BUFSZ - 1, 0); // recebi uma mensagem do client
```

verifico qual comando eu recebi do cliente, podendo ser:

1- INS_REQ (install param ou install file)

```
if (strncmp(requ_1, buf, 7) == 0){ ////////////// recebi INS_REQ do cliente
```

1.1-Retorno para o cliente 2 possíveis saídas.

```
install param 1 2 3 4
successful installation
install param 1 3 4 5
sensor already exists
```

2- REM_REQ (remove)

```
if (strncmp(requ_2, buf, 7) == 0){ ////////////// recebi REM_REQ do cliente
```

2.1-Retorno para o cliente 2 possíveis saídas.

```
remove 3
sensor not installed
remove 1
successful removal
```

3- CH_REQ (change param ou change file)

```
if (strncmp(requ_3, buf, 6) == 0){ ////////////// recebi CH_REQ do cliente
```

3.1-Retorno para o cliente 2 possíveis saídas.

```
change param 2 3 4 5
sensor not installed
change param 1 4 5 6
successful change
```

4- SEN_REQ (show value)

```
if (strncmp(requ_4, buf, 7) == 0){ ////////////// recebi SEN_REQ do cliente
```

4.1-Retorno para o cliente 2 possíveis saídas.

```
show value 1
sensor 1: 20 6
show value 4
sensor not installed
```

5- VAL_REQ (show values)

```
if (strncmp(requ_5, buf, 7) == 0){ ////////////// recebi VAL_REQ do cliente
```

5.1-Retorno para o cliente 2 possíveis saídas.

```
show values
sensors: 1 (20 6) 3 (20 6)
```

```
show values
no sensors
```

6- close (Comando inválido)

```
if (strncmp("close", buf, 5) == 0){ ////////////// recebi close do cliente
```

6.1- Fecha o terminal do cliente.

```
gabriel89067@DESKTOP-7OHVILA:~/programas$ ./client 127.0.0.1 90900
[log] connected to IPv4 127.0.0.1 25364
insta pram 3 4 5 6
gabriel89067@DESKTOP-7OHVILA:~/programas$
```

7- kill (kill)

```
if (strcmp(buf, "kill") == 0){ ////////////// recebi kill do cliente
```

7.1-Fecha o terminal do cliente e do servidor.

```
gabriel89067@DESKTOP-7OHVILA:~/programas$ ./client 127.0.0.1 90900
[log] connected to IPv4 127.0.0.1 25364
kill
gabriel89067@DESKTOP-7OHVILA:~/programas$
```

```
gabriel89067@DESKTOP-7OHVILA:~/programas$ ./server v4 90900
bound to IPv4 0.0.0.0 25364, waiting connections
[log] connection from IPv4 127.0.0.1 49118
gabriel89067@DESKTOP-7OHVILA:~/programas$
```

- **Dados armazenados no servidor**

Armazeno no servidor, uma struct de até 100 sensores e suas características como mostra a figura:

```
struct equipamento
{
    int ID;
    int CORR;
    int TENS;
    int EFI;
    int POT;
};
```

3. ARQUITETURA DO CLIENTE (RTU)

O sistema é baseado em uma arquitetura cliente-servidor, utilizando a comunicação via TCP/IP. O cliente (RTU) é responsável pelo envio das solicitações para o servidor (MTU) e pela gestão dos inputs do teclado.

- **Obrigações do cliente**

1- O cliente recebe comandos do teclado e os manipula para enviar para o servidor.

```
gabriel89067@DESKTOP-7OHVILA:~/programas$ ./client 127.0.0.1 90900
[log] connected to IPv4 127.0.0.1 25364
change param 2 3 4 5
sensor not installed
change param 1 4 5 6
successful change
show value 1
sensor 1: 20 6
vhow value 3
```

- **Funcionalidades do cliente**

Após o recebimento da mensagem via teclado , como mostrado abaixo:

```
memset(pala_in, 0, 50);
fgets(pala_in, 50, stdin);
pala_conhecid = 0;
```

verifico qual comando eu recebi do cliente, podendo ser:

1- install param

```
if (strncmp(pala_1_2, pala_in, 13) == 0){ // digitei install param
```

Comando que requisita ao servidor via código “INS_REQ” a instalação de um sensor por meio de parâmetros passados na tela .Sua sintaxe é da forma: `install param 1 2 3 4` .
Tratamento de 2 possíveis erros :

1.1- erro de comando ,envia ao servidor o código “close” e fecha o terminal do cliente

```
intall param 1 2 3 4
gabriel89067@DESKTOP-7OHVILA:~/programas$
```

1.2- erro de parâmetro , imprime “invalid sensor”

```
install param 1 2 3
invalid sensor
```

2- install file

```
if (strncmp(pala_1_1, pala_in, 12) == 0){ // digitei install file
```

Comando que requisita ao servidor via código “INS_REQ” a instalação de um sensor por meio de um arquivo .txt .Sua sintaxe é da forma: `install file file1.txt` .
Tratamento de 2 possíveis erros :

2.1- erro de comando ,envia ao servidor o código “close” e fecha o terminal do cliente

```
install fil file1.txt
gabriel89067@DESKTOP-7OHVILA:~/programas$
```

2.2- erro de file , imprime “invalid sensor”

```
install file file.txt
invalid sensor
```

3- remove

```
if (strncmp(pala_2, pala_in, 6) == 0){ // digitei remove
```

Comando que requisita ao servidor via código “REM_REQ” a remoção de um sensor por meio de um parâmetro passado na tela.Sua sintaxe é da forma: `remove 2`.
Tratamento de 2 possíveis erros :

3.1- erro de comando ,envia ao servidor o código “close” e fecha o terminal do cliente

```
remov 1
gabriel89067@DESKTOP-7OHVILA:~/programas$
```

3.2- erro de param , imprime “invalid sensor”

```
remove
invalid sensor
```

4- show value

```
if (strncmp(pala_4_1, pala_in, 11) == 0){ // digitei show value
```

Comando que requisita ao servidor via código “SEN_REQ” a visualização de um sensor por meio de um parâmetro passado na tela .Sua sintaxe é da forma: `show value 2`.
Tratamento de 1 possíveis erros :

4.1- erro de comando ,envia ao servidor o código “close” e fecha o terminal do cliente

```
show valu 1
gabriel89067@DESKTOP-7OHVILA:~/programas$
```

5- show values

```
if (strncmp(pala_4_2, pala_in, 11) == 0){ // digitei show values
```

Comando que requisita ao servidor via código “VAL_REQ” a visualização de todos os sensor por meio de um parâmetro passado na tela .Sua sintaxe é da forma: `show values`.
Tratamento de 1 possíveis erros :

5.1- erro de comando ,envia ao servidor o código “close” e fecha o terminal do cliente

```
show vlues
gabriel89067@DESKTOP-7OHVILA:~/programas$
```

5.2- erro de param , imprime “invalid param”

```
show values 3
invalid param
```

6- change param

```
if (strncmp(pala_3_1, pala_in, 12) == 0){ // digitei change param
```

Comando que requisita ao servidor via código “CH_REQ” a modificação de um sensor por meio de parâmetros passados na tela .Sua sintaxe é da forma: `change param 1 2 3 4`
Tratamento de 2 possíveis erros :

6.1- erro de comando ,envia ao servidor o código “close” e fecha o terminal do cliente

```
change parm 1 2 3 4
gabriel89067@DESKTOP-7OHVILA:~/programas$
```

6.2- erro de param , imprime “invalid sensor”

```
change param 1 2 3
invalid sensor
```

7- change file

```
if (strncmp(pala_3_2, pala_in, 11) == 0){ // digitei change file
```

Comando que requisita ao servidor via código “CH_REQ” a modificação de um sensor por meio de um arquivo .txt .Sua sintaxe é da forma: `change file file1.txt` .

Tratamento de 2 possíveis erros :

7.1- erro de comando ,envia ao servidor o código “close” e fecha o terminal do cliente

```
chang file file1.txt
gabriel89067@DESKTOP-7OHVILA:~/programas$
```

7.2- erro de param , imprime “invalid sensor”

```
change file file.txt
invalid sensor
```

8- kill

```
if (strncmp("kill", pala_in, 4) == 0)
```

Comando que requisita ao servidor via código “kill” a instalação de um sensor por meio de um parâmetro passado na tela .Sua sintaxe é da forma: `kill` .