Trabalho Prático

Sistemas Operativos

Departamento de engenharia informática (DEIS)

Daniel Bravo (2021137795)
Daniel Rodrigues (2021142013)

1. Introdução

O trabalho baseia-se num sistema de leilão onde os clientes pedem x e o sistema trata de lhe responder ao pedido, podem lançar itens á venda e poder comprar items.

2. Frontend

No frontend fizemos uma leitura e validação do username e password dado pelo argc e argv, sendo esta a ordem username password, caso um desses parâmetros seja inválido (tamanho da string) ou seja diferente de 3 o argc sai do programa.

A validação do user é enviada para o FIFO do backend, com auxilio da biblioteca dos users é feito a validação e devolve essa informação pelo FIFO do FRONTEND correspondente.

Para comunicação frontend backend é enviado uma estrutura para o FIFO do CLIENT, que depois é lido a resposta enviada pelo BACKEND através do FIFO do FRONTEND.

Temos 2 threads:

Noti – Cria um ficheiro FIFO para cada USER e sempre que um item é vendido/listado recebe essa informação do backend e apresenta essa informação a cada user.

Balcao-Cria um FIFO para cada user que vai avisar o backend que ainda está vivo com um sinal enviado de x em x tempo defenido pela variável de ambiente HEARTBEAT.

3. Backend

No backend é feito uma validação inicial caso seja lido um valor no argc diferente de 1 é dado uma indicação de erro e sai do programa. Feito a validação é pedido ao administrador qual a função que ele queira usar.

O backend trata e lançar promotores, responder aos pedidos do cliente e ao administrador.

Assim que ele é aberto cria um FIFO e faz a verificação se já existe um FIFO para garantir que apenas esteja aberto apenas um backend.

É criado 4 threads:

Menu – onde o administrador pode usar os seus comandos.

Balcao- Cria um FIFO que recebe “sinais” de todos os frontends e gere estes.

Timer- que acede a uma variável global, que serve para mostrar o tempo de “vida” do backend.

t- que é criado após um user se conectar com o backend chama a função atende cliente, envia uma estrutura com os dados necessários para responder e tratar dos comandos pedidos pelo FRONTEND, como tratar das vendas, compras, atualização de saldo, entre outras funções.

Os promotores são lançados por unnamed pipes.

4. Makefile

O makefile tem a opção all que compila todos os programas, o backend compila os seus programas necessários como backend.c e users_lib.o, o frontend compila o frontend.c, e o clean remove todos os executáveis.

```
all: frontend backend

frontend:
    gcc -o frontend frontend.c

backend:
    gcc -o backend backend.c users_lib.o -pthread

clean:
    rm frontend
    rm backend
```

5. Var. Ambiente

Temos 4 variáveis ambiente: FUSERS(nome do ficheiro que contém informação dos users), FITEMS(nome do ficheiro que contém a informação dos items), FPROMOTORS(nome do ficheiro dos promotores) e HEARTBEAT.

6. Estruturas

```c
typedef struct{
    char name[20];
    int pid;

}prom;

typedef struct{
    prom promotor[MAX_PROGPROMOTORES];
    int heartbeat;
    char fpromotores[30];

}backend;
```

```c
struct item
{
    int id_item;
    char nome_item[MAX_CHAR];
    char categoria[MAX_CHAR];
    int valor_base;
    int compra_imediata;
    int tempo;
    char vendedor[USRNAME_MAX];
    char comprador[USRNAME_MAX];
};

struct cliente
{
    char nome[USRNAME_MAX]; //username
    char passwd[PSWDCHAR_MAX];
    bool estado_on;
    int saldo;
};

struct B_F{
    int temp;
    int licitacao,deposito,prec_MAX;
    char comando[MAX_COMAND];
    itens item;
    pid_t cli_pid;
    bool ligado;
    char username[USRNAME_MAX];
    char password[PSWDCHAR_MAX];
    char resposta[100];

    char fitem[30];
    char fileusers[30];
};
```

7. Conclusão

Contudo apesar de ser um trabalho trabalhoso, foi um tema interessante, adquirimos vários conhecimentos que não tinhamos uma vez que é um tipo de programação diferente.