EP1: MAC0422 - Sistemas Operacionais

Thiago Cunha Ferreira e Eduardo Freire de Carvalho Lima

1 Introdução

Este relatório descreve o que foi feito no primeiro EP (Exercício-Programa) da matéria *MAC0422 - Sistemas Operacionais*, onde tivemos que criar uma shell simplificada que executava 4 comandos:

- protegepracaramba <caminho do arquivo>
- liberageral <caminho do arquivo>
- rodeveja <caminho do programa>
- rode <caminho do programa>

A especificação de cada uma pode ser vista no PDF-Enunciado no PACA.

2 Implementação

2.1 Máquina Virtual

A versão de Minix 3 proposta para esse EP foi instalada em uma máquina virtual *VirtualBox* com as especificações sugeridas no enunciado (1024 MB de memória, 512 MB de disco VDI dinamicamente alocado). Adicionalmente, foi instalado o editor de texto *VIM*. A compilação foi feita com o compilador *CC*.

Para acessar a máquina virtual, deve ser feito login com o usuário "root".

2.2 Shell

A shell implementada é bastante simples, construída para suportar os 4 comandos especificados. Por exemplo, deve haver necessariamente duas "palavras" (sequências de caracteres não brancos) para o prosseguimento da shell, sendo que a primeira só suporta até 20 caracteres e a segunda, 400. Após esse número, as palavras serão "cortadas".

Apesar disso, ela avisa quando o usuário digita o comando errado.

Sob certas circunstâncias, o indicador de linha de comando vem precedido de um ">", mas por simplificação da shell, pode haver casos onde ela não indica

realmente a linha de execução ou possui texto vindo da saída padrão (como ao usar a função "rode()"). Em qualquer caso, lembre-se que o que será lido são **as duas palavras vindas da entrada padrão**.

2.3 Funções

Uma descrição das funções implementadas no projeto será apresentada a seguir:

- **protegepracaramba <caminho do arquivo>**

  Implementamos essa função com o comando *chmod()*. Como descrito no enunciado, ela altera a proteção do arquivo para 000, ou seja, o arquivo dado como parâmetro não poderá ser escrito, lido ou executado pelo "user", "group" ou "others". Para utilizarmos o comando *chmod()*, importamos a bilbioteca *<sys/stat.h>* e escrevemos a função *chmod(diretório, "número")*, no qual o número vai de 000 a 777 (cada digito não pode passar de 7). Nesse caso, o parâmetro foi "0000", onde primeiro 0 é indicador de número octal.

- **liberageral <caminho do arquivo>**

  Seguindo a mesma lógica da função *protegepracaramba*, a *liberalgeral*, por sua vez, recebe o parâmetro "0777", permitindo que o "user", o "group" e os "others" tenham permissão de "read", "write" e "execute" no arquivo dado como parâmetro.

- **rodeveja <caminho do programa>**

  A função *rodeveja* recebe como parâmetro um programa a ser executado. Para isso, é realizado um *fork()*, onde o pai espera o processo-filho acabar através da função *wait()* (que suspende as atividades do processo-pai enquanto esse estiver executando). Quando o processo filho acabar sua execução, o código de erro estará inbutido nos 8 últimos bits da variável *status*. Para adquirí-lo, usamos os macros *WIFEXITED(status)* e *WEXITSTATUS(status)*, ambos da biblioteca *sys/wait.h*, que checam se o processo-filho retornou normalmente (através das funções *exit()* ou retornando do *main()*) e devolve o código de erro, respectivamente.

  No caso do filho, realizamos a chamada de função *execve()* para rodar o programa em questão como um novo processo. Se houver algum erro, o *execve()* irá retornar -1 pra nossa variável *n* (inicialmente igual a 0), e o erro em si estará na variável *errno*, encontrada na biblioteca*<errno.h>* . Nisso, atribuímos o *errno* a variável *n* e o devolvemos através da função *exit(n)*, que terminará o processo-filho.

- **rode <caminho do programa>**

  Semelhante ao processo *rodeveja*, o *rode* não precisa suspender o funcionamento do pai, por isso executamos um *fork()* e, quando estiver no filho, um *execve()*. Após o fim do programa, *exit()* para terminar o processo com código 0 (o código de erro específico não é necessário para esta função).

3 Organização e modo de uso

Estando de acordo com as especificações do projeto, o arquivo em C *mac422shell.c* com o código fonte está localizado no diretório */usr/local/src* e o arquivo executável, em */usr/local/bin*, compilado com o comando "cc -o mac422shell mac422shell.c" e transferido para essa pasta.

Para o uso do programa, basta digitar "mac422shell" no terminal do MINIX. Para a sua leitura do código-fonte, acessar seu diretório e abrir arquivo com editor de texto (para abrir com o VIM, usar o comando "vim [NOME DO ARQUIVO]").

4 Outros detalhes

Outras decisões de projeto feitas durante o desenvolvimento:

- Teclado brasileiro instalado;
- Uso do compilador CC, ao invés do GCC, para limitar o tamanho do arquivo final.
- Foi feita apenas a instalação do *vim* das opções presentes no *packman* como editor de texto do nosso projeto. Outros poderão ser adicionados, dependendo da conveniência e da limitação de espaço para envio pelo PACA.

OBS: Há um programa de testes, "hello", que imprime "Hello, World!" na saída padrão que pode ser usado para alguns testes. Ele está em /home.