EXERCÍCIO-PROGRAMA 4

SISTEMAS OPERACIONAIS — MAC0422

RENATO LUI GEH
NUSP: 8536030
GUILHERME FREIRE
NUSP: 7557373

1. Introdução

O EP foi feito em um Minix 3.1.2a simulado pela VM VirtualBox. Os arquivos fonte estão localizados em `/usr/local/`.

Os arquivos modificados foram:

- `/usr/local/include/minix/callnr.h`
- `/usr/local/src/include/minix/callnr.h`
- `/usr/local/src/lib/posix/Makefile.in`
- `/usr/local/src/servers/fs/inode.c`
- `/usr/local/src/servers/fs/inode.h`
- `/usr/local/src/servers/fs/proto.h`
- `/usr/local/src/servers/fs/table.c`
- `/usr/local/src/servers/fs/open.c`

As versões modificadas estão em `/usr/local/`, assim como os arquivos não modificados. Deste jeito, pode-se rodar `/usr/local/src/tools/Makefile` sem alterar o código original. Um arquivo foi adicionado:

- `/usr/local/src/lib/posix/_lsr.c`

Quando os blocos de código transcritos neste relatório não forem muito grandes, vamos indicar as modificações feitas. Um símbolo `-` no início da linha indica a linha original no Minix. Um símbolo `+` no início da linha indica a nova linha adaptada para o EP. Uma linha vazia com o símbolo `-` indica que no código original a linha não existia. Analogamente, `+` em uma linha vazia indica que deletamos a linha original. Um `#` indica um comentário no código, ou seja, a linha indicada por este símbolo não existe no arquivo original.

Date: 11 de dezembro de 2016.

2. lsr()

3. Arquivos imediatos

Neste EP, essa função não foi implementada por dificuldades técnicas.

Inicialmente, procuramos materiais de referência na Internet para nos auxiliar com essa tarefa. Encontramos esse documento *https://minixnitc.github.io/report.pdf* que descreve uma implementação muito próxima do nosso objetivo. Ele foca principalmente na leitura e escrita em arquivos imediatos e seu comportamento.

Procuramos entender o algoritmo descrito, porém ele omite partes importantes, de forma que não foi possível compreender os detalhes, além de ter estruturas que não eram necessárias para o nosso problema. Por esses motivos não foi possível utilizá-lo como guia.

O passo seguinte foi tentar uma implementação partindo do zero. Uma *flag* para apontar que o i-node é um arquivo imediato foi criada no arquivo `inode.h` e inicializada no arquivo `inode.c`. Após isso, procuramos nos outros arquivos do FS onde utilizar essa *flag*, porém não a estrutura de i-node é bastante confusa e não conseguimos implementar essa funcionalidade. Nenhuma tentativa foi bem sucedida, então resolvemos desfazer todas as alterações.

4. Observações importantes quanto a execução do EP

Como usamos um Floppy Controller na nossa VM, a tela inicial irá indicar que não foi encontrado um local de boot. Para resolver isto, pressione `F12` e em seguida pressione `1`. Isto selecionará o controlador principal (a que possue a imagem do Minix) como local de boot.

Como não mudamos o caminho de boot padrão do Minix, quando a VM for rodada, deve-se dar boot na imagem correta. Por padrão, a VM irá dar boot na imagem padrão original.

É recomendável que se recompile o Minix novamente para garantir que tudo esteja o mais recente possível. Caso não se recompile o Minix, a imagem em `/boot/image` mais recente é:

```
/boot/image/3.1.2ar45
```

Para rodar a imagem escolhida, basta indicar o caminho. Por exemplo, caso a imagem desejada seja `/boot/image/3.1.2ar45`, então:

```
# Garanta que esteja na tela de boot.
shutdown
# Indique qual imagem deve ser escolhida.
image=/boot/image/3.1.2ar45
# Faca o boot.
```

```
boot
```