Trabalho II - Sistema de Arquivos T2FS

Relatório Final

Sistemas Operacionais

Arthur Lenz	218316
Guillermo Amaya	217434
Rafael Machado	215020

https://bitbucket.org/Turao/t2fs-returns/

1. Status de desenvolvimento:

Status	Função	Implementação
	identify2	OK
	create2	OK
	delete2	OK
	open2	OK
	close2	OK
	read2	OK
	write2	Parcialmente funcional
	truncate2	Parcialmente funcional
	seek2	OK
	mkdir2	OK
	rmdir2	OK
	opendir2	OK
	readdir2	OK
	closedir	OK

2. Testes realizados:

writing.c

Deseja-se criar um arquivo, escrever dados, e então realizar o fechamento do mesmo sem que ocorra perda dos dados. Após o fechamento, realizamos novamente uma abertura de arquivo, seguida da leitura dos dados.
Caso não ocorra uma inconsistência (esperado), será impresso o conteúdo escrito no início do teste pela função *write2*.

Funções testadas:	
open2	*write2*
close2	*read2*

OBS:
Não conseguimos implementar a gravação da entrada do arquivo de volta para o disco. Logo, temos uma inconsistência em atributos como *bytesFileSize*, essenciais para a leitura do dado novamente - quando necessária releitura do disco (a estrutura em memória mantém a consistência, sendo possível múltiplas leituras desde que não ocorra o fechamento do arquivo). Tanto a escrita dos dados, quanto alocação dos blocos necessários e mapeamento no registro MFT são realizadas com sucesso.

Para tal, é possível analisar os dados do disco através de ferramentas como editores hexadecimais.

dirception.c

Deseja-se testar a criação de múltiplas pastas dado um caminho (que pode ser tanto relativo quanto absoluto).
Na implementação atual, a função mkdir cria todas as pastas não existentes no caminho dado (similar ao comando mkdir -p do unix).
Após a criação dos diretórios, realizamos a impressão das entradas de cada pasta criada.

Funções testadas:	
mkdir2	*readdir2*
opendir2	*closedir2*

bad_developer_directories.c

Simulamos o uso incorreto das funções de diretório:

- criação de diretório quando há conflito de nomes com as entradas do cwd (current working directory)
- remoção de diretório não existente
- abertura de diretório não existente (handle inválido)
- fechamento de diretório não existente (handle inválido)
- remoção do diretório root

Funções testadas:	
mkdir2	*readdir2*
opendir2	*closedir2*

bad_developer_files.c

Simulamos o uso incorreto das funções de arquivo:

- criação de arquivo quando há conflito de nomes com as entradas do cwd (current working directory)
- remoção de arquivo não existente
- abertura de arquivo não existente (handle inválido)
- fechamento de arquivo não existente (handle inválido)
- seek com handle inválido
- seek em posição inválida

Funções testadas:	
create2	*close2*
open2	*seek2*

3. Observações:

Devido ao desenvolvimento ocorrer primariamente em máquinas 64-bit, os Makefiles possuem uma flag (-m32) que forçam a compilação dos códigos fontes para 32-bit.
Em alguns laboratórios **NÃO** foi possível utilizar esta flag. As máquinas **NÃO** suportam esta flag, pois são de 64-bit e não possuem a lib para tal compilação forçada (*libc6-dev-i386*).

Sendo assim, a linkagem com os objetos fornecidos (*apidisk.o* e *bitmap2.o*) **NÃO** são possíveis, impossibilitando os testes nestas máquinas.