Disciplina = Sistemas Operacionais – S12 Data = 03/09/2023

Aluno = Pedro Henrique Grossi da Silva Professor = Luiz Fernando Copetti

Relatório laboratório 3 - Processos

Os arquivos fontes não foram modificados com exceção do fork-execve.c onde foi editada uma linha a fim de forçar um erro de path e ver qual o retorno. Mas isso foi evidenciado no relatório.

Código fork.c

EXPLICAÇÃO DO CÓDIGO

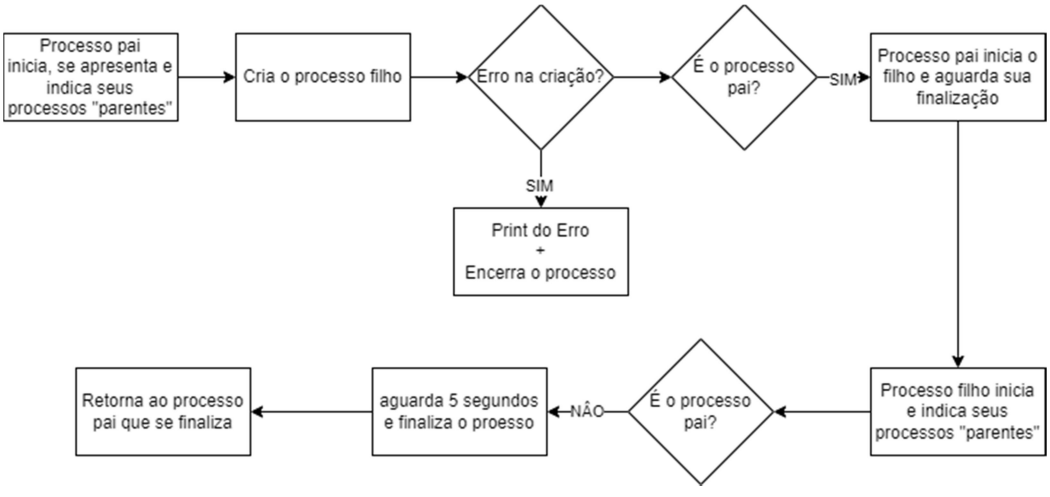
Neste código o processo pai é iniciado e exibe sue ID. Após isso ele inicia um processo filho (retval = fork();).

Após isso ele passa por uma verificação:

- Ocorreu algum problema na criação do processo filho? Exibir “Erro”
- Caso contrario:
  - Caos o processo sendo executado for o pai, aguardar a execução do processo filho.
    - Nisso é iniciado o processo filho. Que realiza o mesmo código.
  - Caso contrario:
    - O processo filho aguarda 5 segundos e se encerra sozinho. Retornando ao processo pai que se finaliza.

```
debian@debian:~/Documents/Lab2$ ./fork
Ola, sou o processo  4945
[retval:  4946] sou  4945, filho de  3270
[retval:     0] sou  4946, filho de  4945
Tchau de  4946!
Tchau de  4945!
debian@debian:~/Documents/Lab2$
```

FLUXO DE TEMPO DO CÓDIGO



Código fork-execve.c

EXPLICAÇÃO DO CÓDIGO

Neste código o processo pai é iniciado e exibe sue ID. Após isso ele inicia um processo filho (retval = fork();).

Após isso ele passa por uma verificação:

- Ocorreu algum problema na criação do processo filho? Exibir “Erro”
- Caso contrario:
  - Caos o processo sendo executado for o pai, aguardar a execução do processo filho.
    - Nisso é iniciado o processo filho.
  - Caso contrário:
    - O processo filho chama a execução dissociada de outro executável. No caso como o processo pai estar esperando o processo filho terminar (wait(0);) ele não se encerra mas a execução é dissociada.

```
debian@debian:~/Documents/Lab2$ ./fork-execve
Ola, sou o processo  4981
[retval:  4982] sou  4981, filho de  3270
[retval:     0] sou  4982, filho de  4981
Sun Sep  3 11:31:18 AM -03 2023
Tchau de  4981!
debian@debian:~/Documents/Lab2$
```

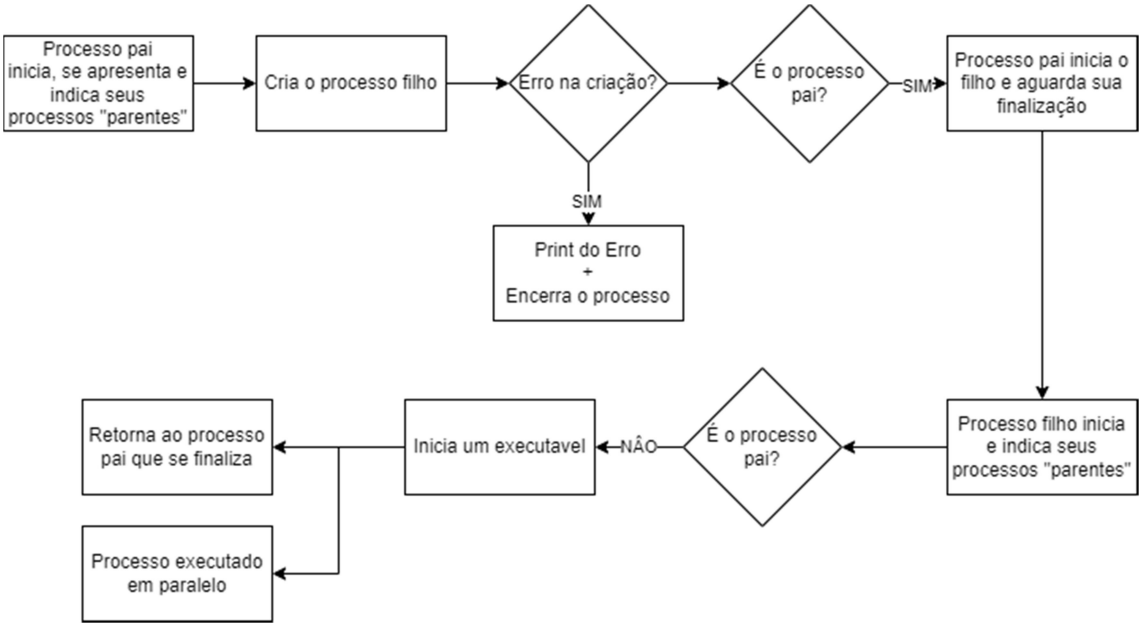
Caso o executável chamado na função execve não existir é retornado um erro (“Erro: No such file or directory”) e o processo se encerra.

(Nesse exemplo foi alterado a linha de código abaixo para forçar o erro)

```
execve ("/bin/dat", argv, envp) ; // processo filho se dissocia do pai
```

```
debian@debian:~/Documents/Lab2$ ./fork-execve
Ola, sou o processo  5017
[retval:  5018] sou  5017, filho de  3270
[retval:     0] sou  5018, filho de  5017
Erro: No such file or directory
Tchau de  5018!
Tchau de  5017!
debian@debian:~/Documents/Lab2$
```

FLUXO DE TEMPO DO CÓDIGO



Código fork-print.c

EXPLICAÇÃO DO CÓDIGO

O código inicia com o processo pai abrindo um processo filho, ai ele exibe seu ID e o valor de x (no caso é ZERO).

Após isso ele passa por uma verificação:

- Ocorreu algum problema na criação do processo filho? Exibir “Erro”
- Caso contrario:
  - Caos o processo sendo executado for o pai, x = 0 e aguardar a execução do processo filho.
    - O processo filho é iniciado executando o mesmo código do pai.
  - Caso contrario:
    - X++; e espera por 5 segundos.
    - Após isso o programa filho se finaliza exibindo o valor de x (no caso é igual a 1) e retorna ao processo pai que se finaliza exibindo o valor de x (no caso é ZERO).

Essa diferença de valores acontece, pois o valor de x foi modificado apenas na condição do processo filho, logo ele foi modificado apenas no processo filho. Quando houve o retorno para o processo pai o x (do processo pai) não havia sido modificado.

```
debian@debian:~/Documents/Lab2$ ./fork-print
No processo  5194 x vale 0
No processo  5195 x vale 0
No processo  5195 x vale 1
No processo  5194 x vale 0
```

FLUXO DE TEMPO DO CÓDIGO