Sistemas Operacionais

Criação e Encerramento de Processos

Lesandro Ponciano

2025

Objetivos da Aula

- Apresentar conceitos básicos associados ao
  - Criação e encerramento de processos
  - Árvore de processos
  - Ataque de negação de serviço baseado em criação de processos

Criação de Processos

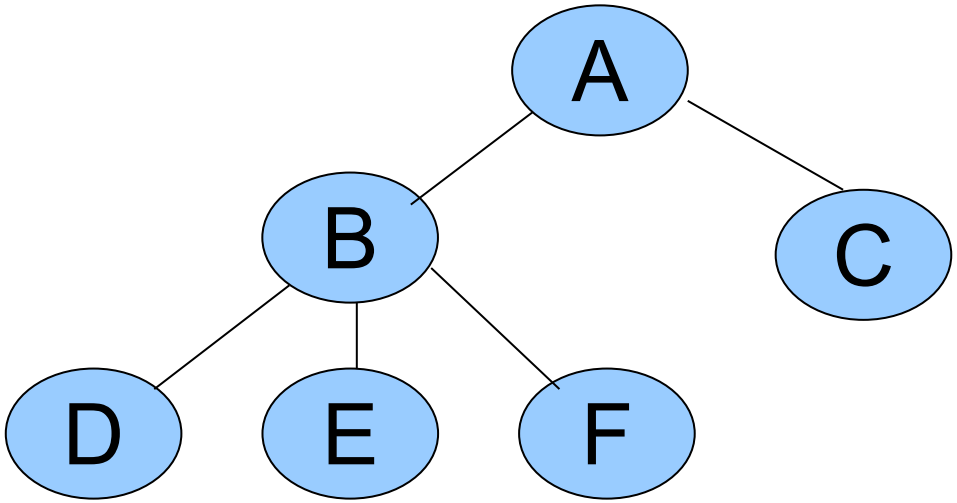
- Cada processo é identificado por um *id* de processo
  - Identificador de processo *(process identifier, pid)*

- Processos podem ser
  - executados concorrentemente
  - criados e excluídos dinamicamente

Processos Pai e Filho

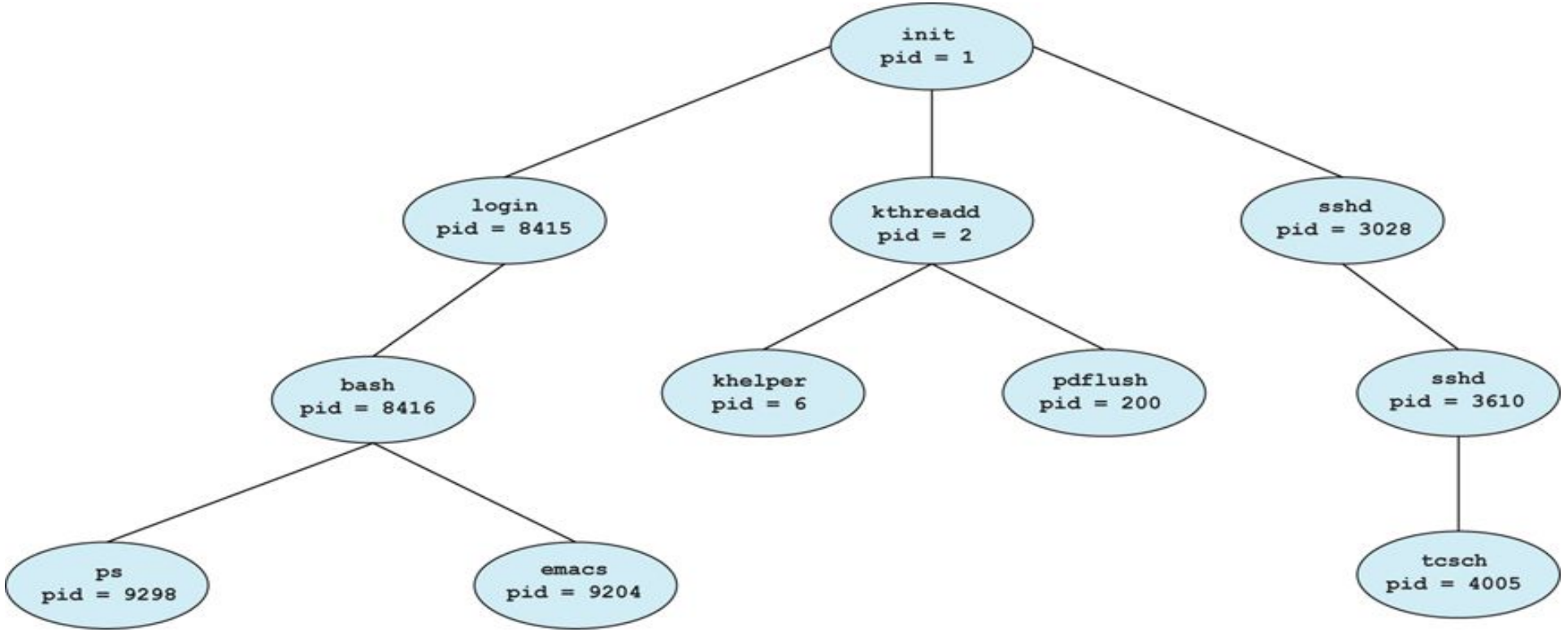
- Processos podem criar outros processos
  - O criador é chamado pai
  - Os novos processos são chamados filhos
- Filhos podem criar outros filhos originando uma árvore

Árvore de Processos



- Pais e Filhos
  - O processo *A* criou os processos *B* e *C*
  - O processo *B* criou os processos *D*, *E* e *F*

Árvore de Processos do Linux



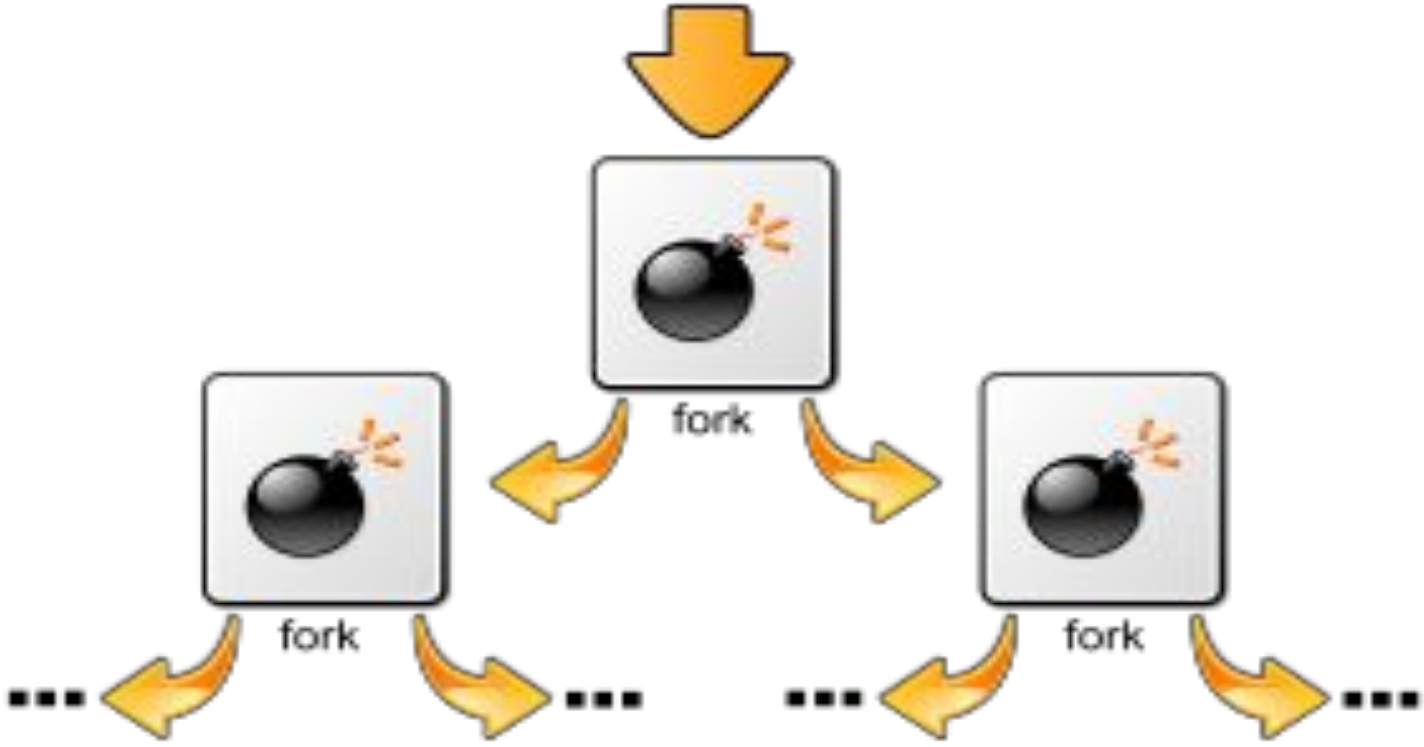
Processos Pai e Filhos

- Filhos possuem PCB, contador, registros, *pid*
- Pais conhecem a identidade (*pid*) dos filhos
- Processos filhos podem usar recursos do pai ou podem requerer recursos ao SO
  - Restringir que um filho use recursos do pai impede que um processo sobrecarregue o sistema criando filhos demais

Fork Bomb

- *“Rabbit vírus”* (1969)
  - Ataque de negação de serviço baseado na criação de processos
- Um processo que cria novos processos até sobrecarregar completamente o sistema
  - CPU, memória, tabela de processos
- Solução: limitar o número de processos que podem ser criados pelo usuário

Fork Bomb



Execução e Endereçamento

- 2 possibilidades em termos de execução
  - O pai continua executando concorrentemente com seus filhos
  - O pai espera até que algum de seus filhos ou todos eles sejam encerrados

- 2 possibilidades em termos de espaço de endereçamento
  - O filho é uma duplicata do pai (possuí o mesmo programa e dados); chamada *fork* no UNIX
  - O filho tem um novo programa carregado nele; chamada *exec* no UNIX

Encerramento de Processos

- O processo é encerrado quando
  - termina a execução de sua última instrução e solicita ao SO que o exclua (*exit()*)
  - outro processo, do mesmo usuário, solicita o encerramento
  - o pai encerra os seus filhos
- Ao encerrar um processo, o SO desaloca todos os seus recursos
- Alguns sistemas implementam cascata pai-filhos
  - Se o pai é encerrado, todos os seus filhos também são

Encerramento de Processos

```
using System;
using System.Diagnostics;
public class Processos
{
    static void Main(string[] args)
    {
        //Cria um processo do programa notepad (bloco de notas)
        Process proc = Process.Start("notepad");

        //Pega o pid do processo criado
        Console.WriteLine("O processo que criei é: "+ proc.Id);

        Console.ReadKey();
        //Mata o processo
        proc.Kill();

        Console.WriteLine("Agora ele está morto: " + proc.Id);
        Console.ReadKey();
    }
}
```

O que este código faz?

```
using System.Diagnostics;
public class Processos
{
    static void Main(string[] args)
    {
        while (true)
        {
            Process proc = Process.Start("notepad");
        }
    }
}
```

O que este código faz?

```
using System;
using System.Diagnostics;
public class Processos
{
    static void Main(string[] args)
    {
        /*
           O método GetProcessesByName retorna uma lista de processos que
           correspondem ao nome especificado como argumento, i.e "notepad"
        */
        Process[] processos = Process.GetProcessesByName("notepad");
        foreach (Process processo in processos)
        {
            processo.Kill();
            Console.WriteLine("Está morto: " + processo.Id);
            Console.ReadKey();
        }
    }
}
```

Atividade de Fixação

O que é incorreto sobre a árvore de processos

A) Processo pai conhece o identificador dos processos filhos e pode encerrará-los.
B) Processos filhos podem compartilhar o espaço de endereçamento do processo pai.
C) Ao encerrar o processo pai, pode-se, em cascata, encerrar os processos filhos e seus descendentes.
D) Processos filhos executam concorrentemente e compartilham o PCB do processo pai.

Referências

TANENBAUM, Andrew S. Sistemas operacionais modernos. 3. ed. São Paulo: Pearson Prentice Hall, 2009. xvi, 653 p. ISBN 9788576052371

SILBERSCHATZ, Abraham; GALVIN, Peter B.; GAGNE, Greg. Fundamentos de sistemas operacionais: princípios básicos. Rio de Janeiro, RJ: LTC, 2013. xvi, 432 p. ISBN 9788521622055

Sistemas Operacionais

Prof. Dr. Lesandro Ponciano

https://orcid.org/0000-0002-5724-0094