Visão Geral sobre o Linux

Prof. Jean M. Laine
jean.laine@mackenzie.br

O que é um sistema operacional?

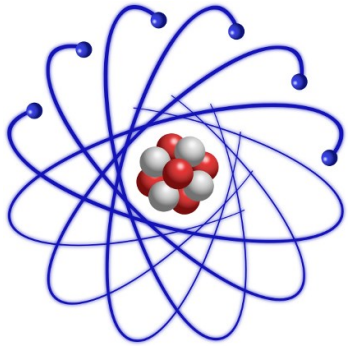
É um programa ou um conjunto de programas cuja função é gerenciar os recursos do sistema, de modo eficiente, fornecendo uma interface amigável entre o computador e o usuário e as aplicações.

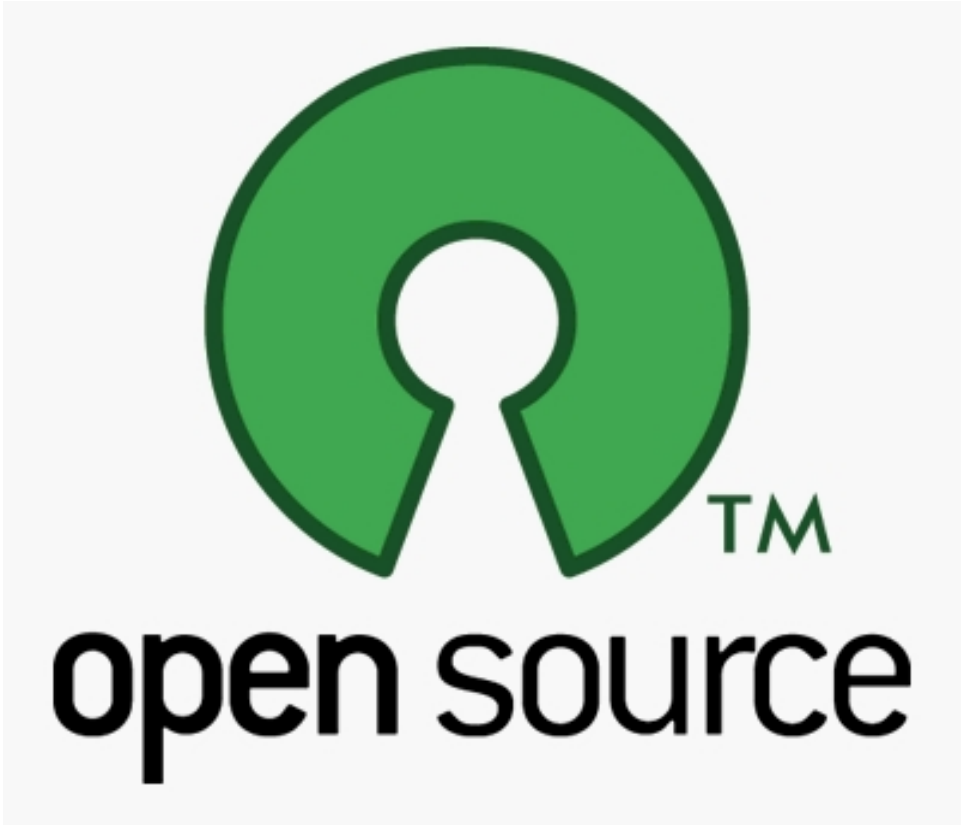
Criatura e seu criador

- A História do Linux começou em **1991** com o início de um projeto pessoal de um estudante finlandês chamado **Linus Torvalds**.
- Inicialmente era só um emulador de terminal para acesso aos servidores UNIX da faculdade
- Baseado no MINIX (1987)
- Outro SO importante na época: BSD (Berkeley Software Distribution) - 1977
- Por que Linux?
  - História da visita ao Zoológico & Aquário Nacional de Camberra, Austrália

Tux
(Torvalds' UniX)

Distribuições





GNU GPL (GNU General Public License, ou Licença Pública Geral GNU), que garante que ele vai ser sempre gratuito e de código aberto.

O que isso significa?

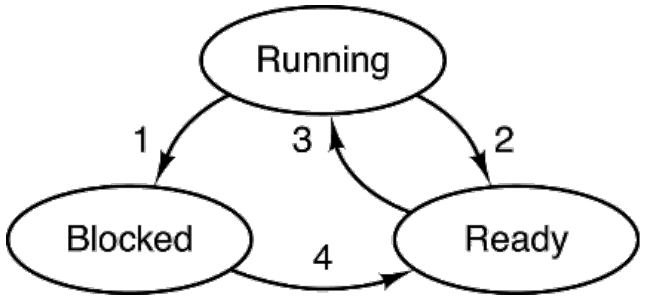
Como um sistema operacional aberto, o Linux é desenvolvido colaborativamente!

Nenhuma empresa é unicamente responsável pelo seu desenvolvimento ou pelo seu suporte continuo.

Seguro, estável e gratuito!

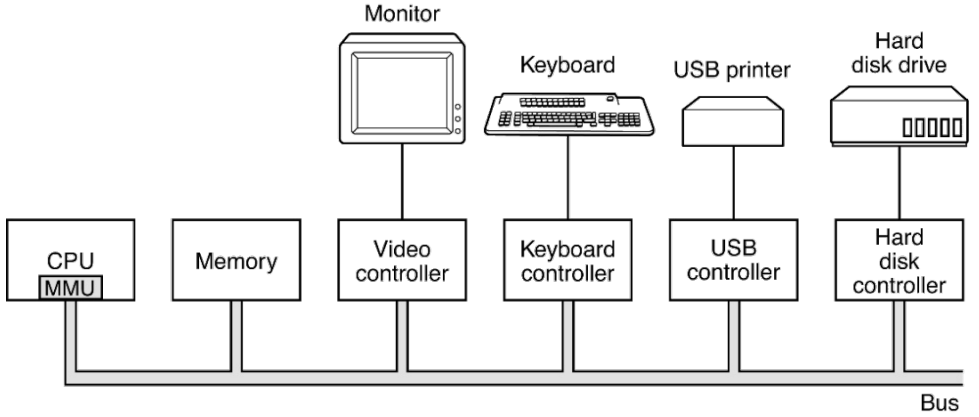
Sistema multitarefa

- Linux é um sistema que consegue executar várias tarefas (processos) ao mesmo tempo
  - Algoritmos de Escalonamento



1. Process blocks for input
2. Scheduler picks another process
3. Scheduler picks this process
4. Input becomes available

- O que acontece quando disparamos uma aplicação/programa?
  - Pensando no hardware



Sistema multiusuário

- Mais de um usuário pode acessar o computador ao mesmo tempo!
  - Uso de terminais virtuais
  - Acesso remoto
  - Cada usuário terá sua área e um controle de acesso
- Tipos de usuário
  - root (superusuário/administrador)
  - comuns
- Linux faz uso de memória virtual
  - O que é memória virtual? Para que serve?
- Sistema de arquivos
  - ext, ext2, ext3, jfs, xfs, ...

Estrutura dos diretórios do Linux

/ diretório raiz;

/boot Kernel do Sistema;

/proc Sistema de arquivos virtual de informação do kernel

/dev Arquivos de dispositivo de hardware

/tmp Arquivos temporários

/etc Arquivos de configuração do sistema

/bin Comandos essenciais do sistema

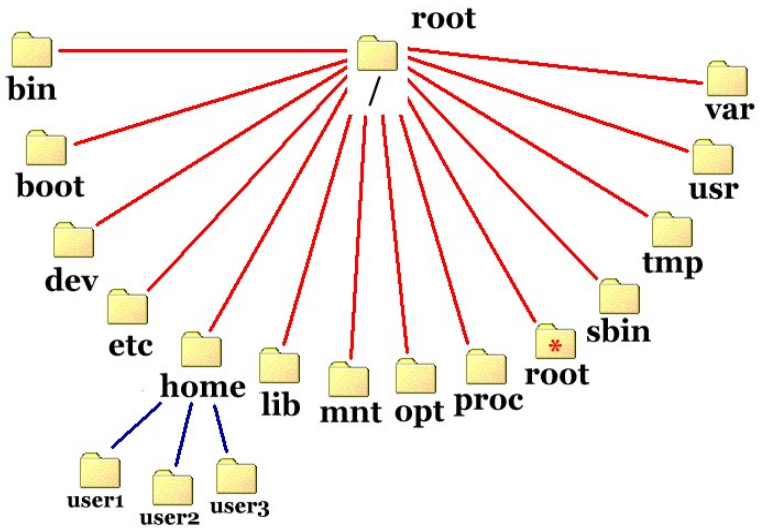
/mnt Ponto de montagem temporário para sistemas de arq.

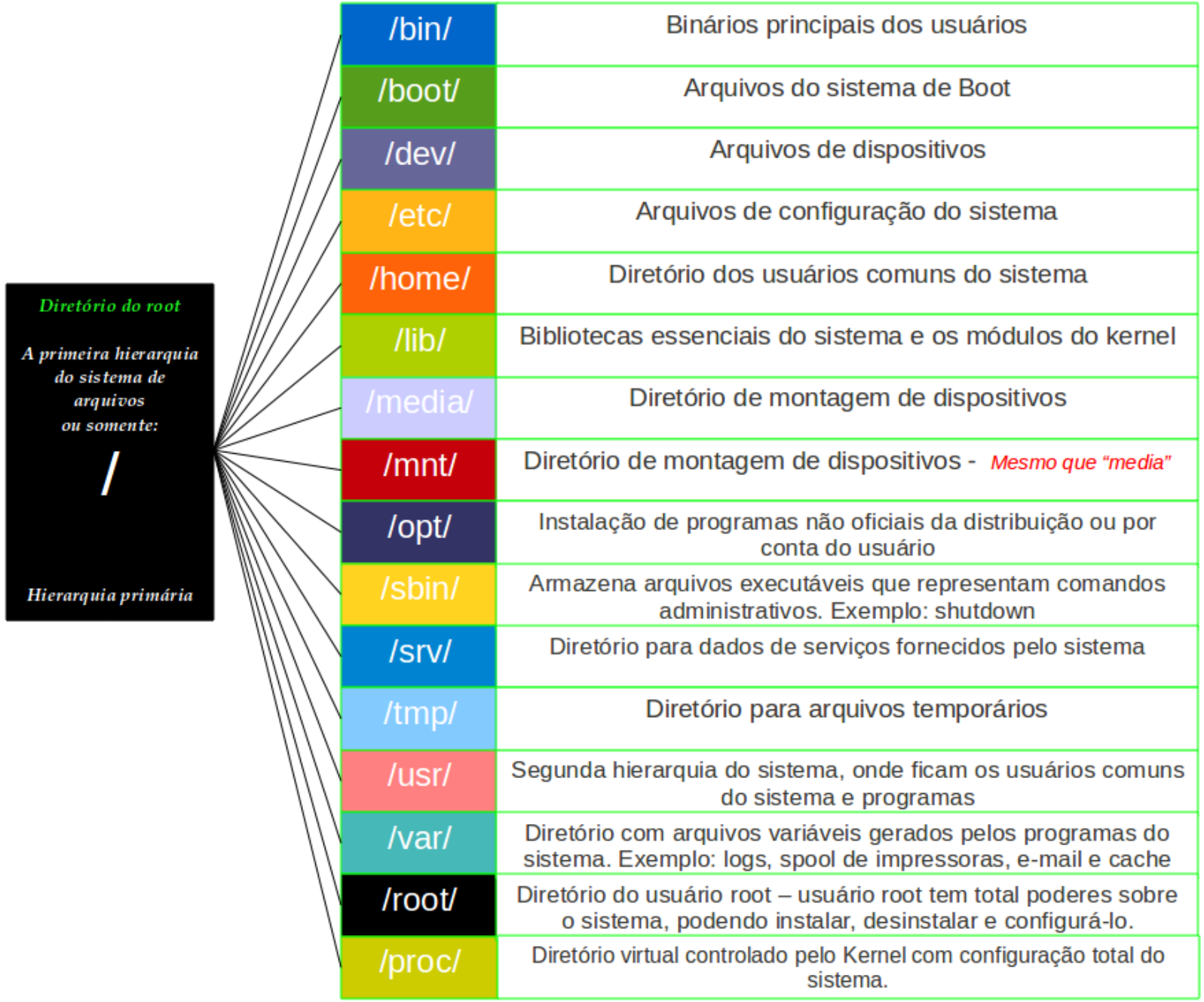
/opt Pacotes de software adicional

/sbin Comandos essenciais de adm. do sistema;

/var Dados variáveis;

/home Diretório do usuário



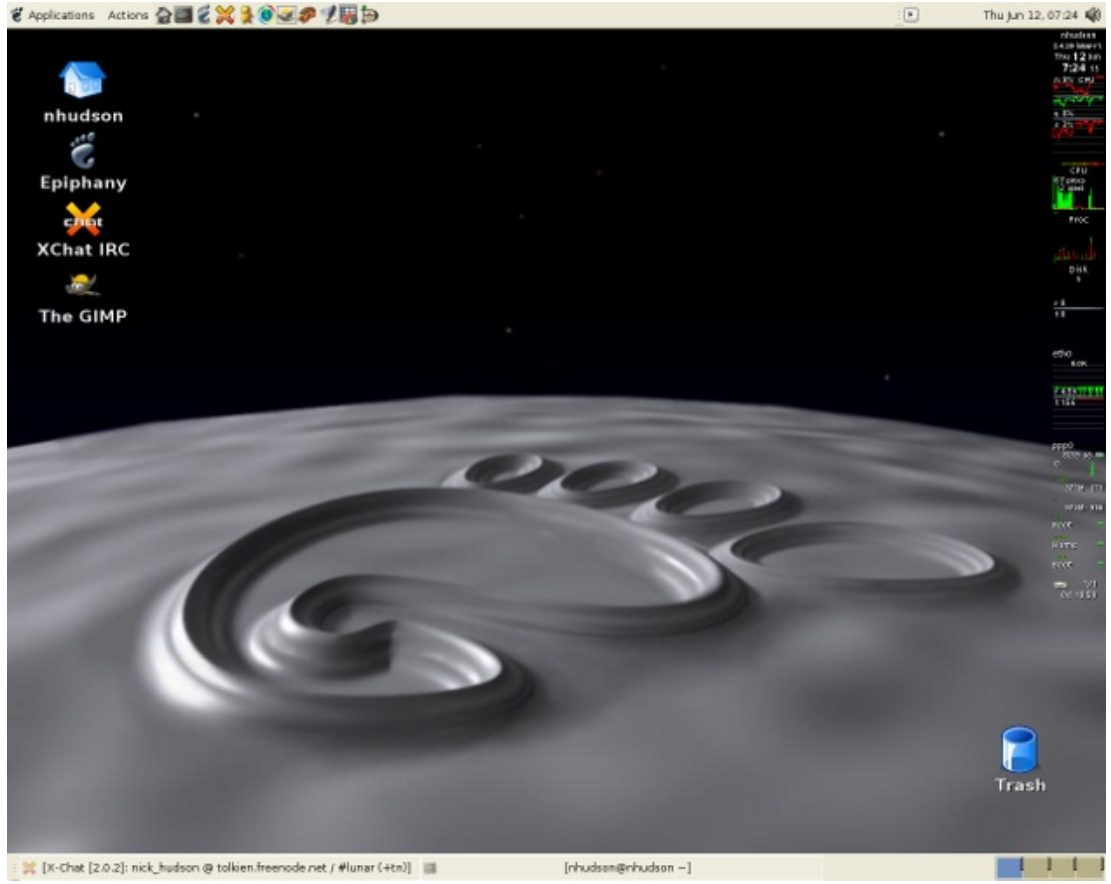

Diretório do root
A primeira hierarquia do sistema de arquivos ou somente:
/
Hierarquia primária
/bin/
Binários principais dos usuários
/boot/
Arquivos do sistema de Boot
/dev/
Arquivos de dispositivos
/etc/
Arquivos de configuração do sistema
/home/
Diretório dos usuários comuns do sistema
/lib/
Bibliotecas essenciais do sistema e os módulos do kernel
/media/
Diretório de montagem de dispositivos
/mnt/
Diretório de montagem de dispositivos - Mesmo que "media"
/opt/
Instalação de programas não oficiais da distribuição ou por conta do usuário
/sbin/
Armazena arquivos executáveis que representam comandos administrativos. Exemplo: shutdown
/srv/
Diretório para dados de serviços fornecidos pelo sistema
/tmp/
Diretório para arquivos temporários
/usr/
Segunda hierarquia do sistema, onde ficam os usuários comuns do sistema e programas
/var/
Diretório com arquivos variáveis gerados pelos programas do sistema. Exemplo: logs, spool de impressoras, e-mail e cache
/root/
Diretório do usuário root – usuário root tem total poderes sobre o sistema, podendo instalar, desinstalar e configurá-lo.
/proc/
Diretório virtual controlado pelo Kernel com configuração total do sistema.

Para pensar... Onde estão usando Linux?

- IoT (Internet das coisas) - Flexibilidade
- Google (Desempenho)
  - Servidores e Android
- Facebook (Segurança)
  - Mais de 2,5 bilhões de usuários
- Twitter
- Amazon (Estabilidade e segurança)
- Bolsa de valores de Nova York – NYSE (Desempenho e Segurança)
- NASA
- IBM
- McDonalds (Ubuntu)
- Submarinos americados (Red Hat)
- SmartWatchs
- Dispositivos móveis: kindle, tablets, smartphones, ...
- Raspberry Pi
- Sistemas de Controle de tráfego aéreo
- Trens bala japoneses
- Tianhe-2
- Servidores para Hospedagem (70%)
- Hackers
- Pizza Hut
- Olx
- Eletroeletronicos
- Carros

E agora?

Gnome


Applications
Actions
Thu Jun 12, 07:24
nhudson
Epiphany
XChat IRC
The GIMP
Trash
[X-Chat [2.0.2]: nick_hudson @ tolkien.freenode.net / #lunar (+tn)]
[nhudson@nhudson ~]


Activities
Tue 10:20
pplware
Windows
Applications
Type to search...
Additional Drivers
AisleRiot Solitaire
Banshee
Brasero
Broadcast Accounts
Broadcast Preferen...
Calculator
Character Map
Computer Janitor
Disk Usage Analyzer
Disk Utility
Email Settings
All
Accessories
Games
Graphics
Internet
Office
Other
Programming
Sound & Video
System Tools
Universal Access

KDE


Pasta da área de trabalho
TIL
monografia
Pasta do usuário
sethost
tamo
testesblog
aula-01.pdf
aula-02.pdf
aula-03.pdf
CPU
RAM
Rede
Temperatura
27°C
qui
sex
sáb

LXDE


My Documents
Install
LXDE
Lightweight X11 Desktop Environment

Dicas para o aprendizado

- Digite qualquer comando seguido de –help (Dois traços e a palavra help) para ver uma descrição detalhada do comando.

EX: wget –help

- Outra forma de conseguir documentação oficial dos comandos Linux é através do comando man (manual). Digite man seguido do nome do comando que você precisa de informação.

EX: man wget

- Se por algum motivo você preferir guardar as informações do man (Manual) de algum comando em um pendrive ou smartphone em PDF para estudos complementares utilize este comando

EX: man -t wget | ps2pdf – wget.pdf

Linux Quick Reference

SOME USEFUL COMMANDS

File/Directory Basics

ls	List files
cp	Copy files
mv	Rename files
rm	Delete files
ln	Link files
cd	Change directory
pwd	Print current directory name
mkdir	Create directory
rmdir	Delete directory

File Viewing

cat	View files
less	Page through files
head	View file beginning
tail	View file ending
nl	Number lines
od	View binary data
xxd	View binary data
gv	View Postscript/PDF files
xdvi	View TeX DVI files

Linux Quick Reference

SOME USEFUL COMMANDS

File Properties

stat	Display file attributes
wc	Count bytes/words/lines
du	Measure disk usage
file	Identify file types
touch	Change file timestamps
chown	Change file owner
chgrp	Change file group
chmod	Change file protections
chattr	Change advanced file attributes
lsattr	List advanced file attributes

File Compression

gzip	Compress files (GNU Zip)
compress	Compress files (Unix)
bzip2	Compress files (BZip2)
zip	Compress files (Windows Zip)

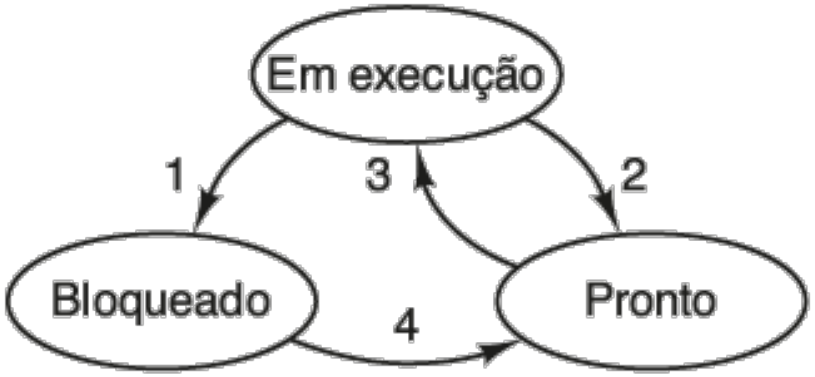
File Comparison

diff	Compare files line by line
comm	Compare sorted files
cmp	Compare files byte by byte
md5sum	Compute checksums

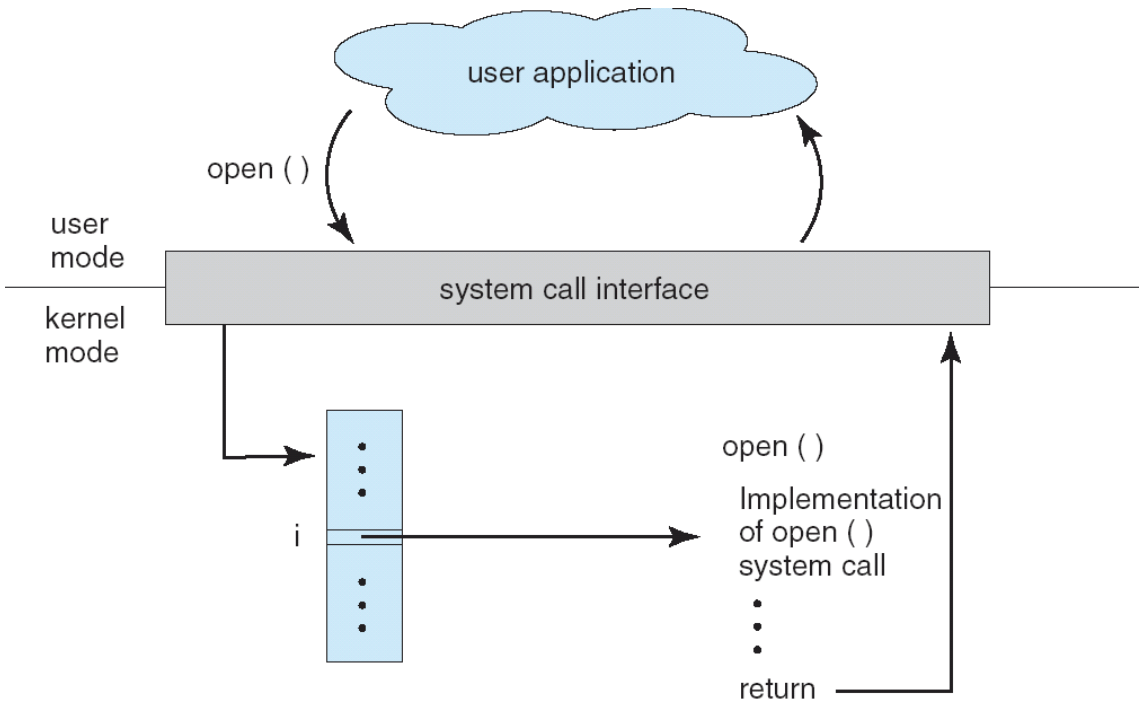
Processos no Linux

- O que é um processo?
  - uma abstração de um programa em execução
- Cada processo tem seu próprio espaço de endereçamento

- Diagrama de Estados



Chamada de Sistema



	Windows	Unix
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
File Manipulation	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Device Manipulation	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shmget() mmap()
Protection	SetFileSecurity() InitializeSecurityDescriptor() SetSecurityDescriptorGroup()	chmod() umask() chown()

Fork ( )

- Criar um processo filho
- Este processo será idêntico ao pai, inclusive tendo as mesmas variáveis, registros, descritores de arquivos, etc.
- Ele vai ser executado e o que acontece em um processo não ocorre no outro, são processos distintos agora
- O que isso significa?
  - É possível mudar o valor de uma variável em um e isso não irá alterar o valor desta variável no outro processo.

Exemplo

```
#include<unistd.h>
#include<stdio.h>
#include<string.h>
#include<errno.h>
int main(){
pid_t filho;
filho = fork();
if (filho == -1) //erro
	printf("Erro fork() %d: %s\n", errno, strerror(errno));
else{
	if (filho == 0) //filho
		printf("Processo-filho: PID = %ld, PPID = %ld\n", (long)getpid(), (long)getppid());
	else //pai
		printf("Processo-pai: PID = %ld, PPID = %ld, PID do filho = %ld\n", (long)getpid(), (long)getppid(), (long)filho);
}
}
```

Atividade

1. Edite, compile e rode o programa exemplo.
   a) use algum editor de texto Linux (Kate, gedit, etc). Nomei o arquivo como fork.c
   b) use o terminal para **compilar**:
      - gcc fork.c -o fork
   c) use o terminal para **executar**:
      - ./fork (enter)
   d) analise a saída do programa
2. Modifique o programa e inclua a declaração de uma variável inteira a com valor 10, antes da chamada fork( ). Depois, faça ambos, processo pai e processo filho, imprimirem o valor desta variável a e verifique se o valor é o mesmo.
3. Altere o valor da variável em algum dos processos e depois imprima. Compare a saída com o que foi apresentado no item 2.