

## Gerenciamento Linux (básico)

Conceitos básicos de sistemas operacionais  
e de sistemas UNIX

Programa de Formação Tecnológica DELL-UFRGS  
<http://www.inf.ufrgs.br/~asc/linux>



## Roteiro

- Conceitos Básicos de Sistemas Operacionais
- Gerência de Processos
- Sistemas de Arquivos
- Sistemas de Entrada e Saída

## Sistema operacional

- Tornar ao usuário a utilização do computador mais conveniente
  - Esconde detalhes internos
  - Reduz o tempo necessário a construção de programas
- Utilizar o hardware do computador de forma eficiente
  - Significa mais trabalho obtido pelo mesmo hardware
  - Obtida por uma melhor distribuição/uso dos recursos

## Diferentes imagens do sistema operacional

- Visão do usuário final:
  - Interpretador de comandos e/ou sistema de janelas
- Visão do programador:
  - Chamadas de sistemas
- Visão do projetista do sistema operacional:
  - Implementação de serviços
- Um administrador de sistemas deve "enxergar" das três formas

## Principais serviços do sistema operacional

---

- Gerência de processo
- Gerência de memória
- Sistema de arquivos
- Sistema de entrada e saída

## Roteiro

---

- Conceitos Básicos de Sistemas Operacionais
- Gerência de Processos
- Sistemas de Arquivos
- Sistemas de Entrada e Saída
- Comandos UNIX "hiper-básicos"

## O conceito de processo (1)

---

- Diferenciação entre o programa e sua execução
- Programa:
  - Entidade estática e permanente
    - Seqüência de instruções
    - Passivo sob o ponto de vista do sistema operacional
- Processo:
  - Entidade dinâmica e efêmera
    - Altera seu estado a medida que avança sua execução
  - Composto por código, dados e contexto (valores em determinado momento)
  - Identificado por um número único (PID - *Process Identifier*)

## O conceito de processo (2)

---

- Abstração que representa um programa em execução
- Diferentes instâncias:
  - Um mesmo programa pode ter várias instâncias em execução, i.e., diferentes processos
  - Mesmo código (programa) porém dados e momentos de execução (contexto) diferentes
- Forma pela qual o sistema operacional "enxerga" um programa e possibilita sua execução
- Processos executam:
  - Programas de usuários
  - Programas de sistema (*daemons*)

## Estados de um processo

### ■ Processos são:

#### ■ Criados

- Momento da execução
- Chamadas de sistemas
- Podem ser associados a uma sessão de trabalho
  - e.g. login + senha → shell (processo)

#### ■ Destruídos

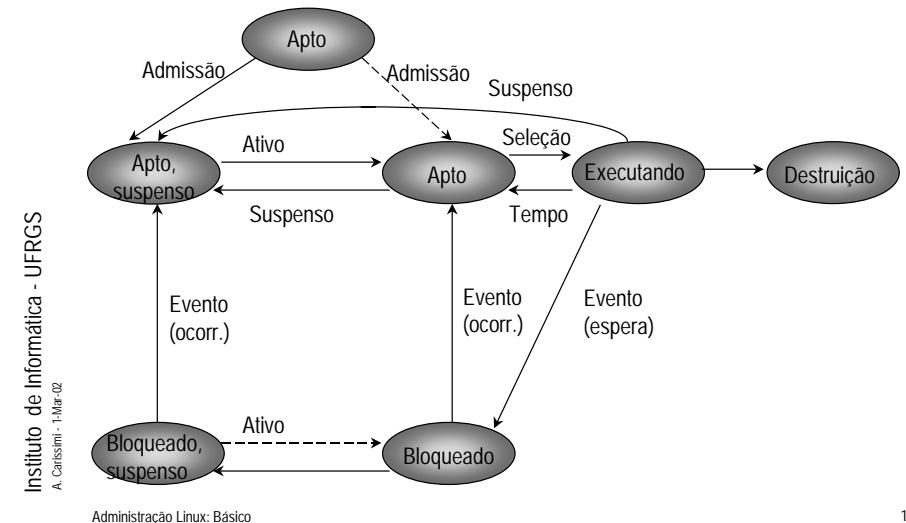
- Término da execução
- Por outros processos

### ■ Entre a criação e a destruição o processo em execução passa por diferentes etapas

#### ■ Estados de um processo

1-9

## Diagrama de estados de processos



Administração Linux: Básico

1-10

## Processos em *foreground* e *background*

### ■ Um processo pode ser executado em *foreground* ou em *background*

### ■ Existe apenas um processo em *foreground* a cada momento

- Processo que interage com o teclado e com tela

### ■ Processos em *background* executam "paralelamente" a outros processos

- Utilizados para executar uma tarefa enquanto se realiza outra
- Disparados utilizando-se o caractere "&"
  - Exemplo: `nedit &`

1-11

## Inicialização do sistema operacional (UNIX-like)

### ■ Um programa de *boot* lê e carrega em memória o núcleo do sistema operacional

### ■ Sistema operacional recebe o controle e realiza:

- Inicialização de tabelas internas de dados
- Inicialização e instalação de drivers de dispositivos
- Criação de um processo especial (*init*) a partir do qual todos os outros processos serão derivados

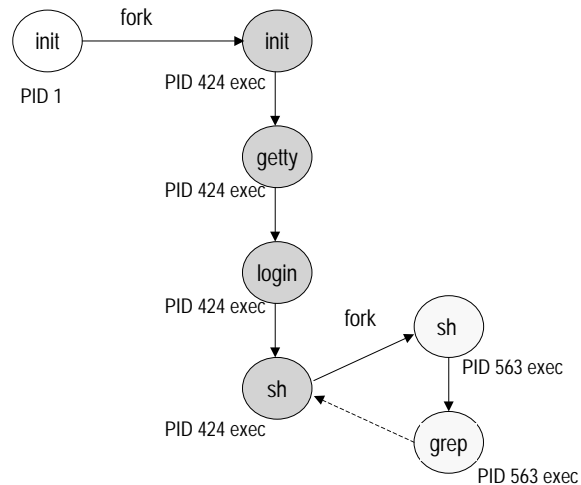
### ■ Conjunto de arquivos de inicialização e configuração (*scripts*)

- Executados pelos processo `init`
- Ordem e localização desses scripts depende do sistema (System V ou BSD)
- Importante conhecer (veremos mais adiante!)

Administração Linux: Básico

1-12

## Criação de processos UNIX: *fork* e *exec*



## Roteiro

- Conceitos Básicos de Sistemas Operacionais
- Gerência de Processos
- Sistemas de Arquivos
- Sistemas de Entrada e Saída
- Comandos UNIX "hiper-básicos"

## Sistema de arquivos

- Parte mais visível do sistema operacional
- Cria um recurso lógico a partir de recursos físicos através de uma interface coerente e simples, fácil de usar
- Mecanismo para armazenamento e acesso a dados e a programas
- Duas partes básicas:
  - Arquivos
    - Armazenamento de dados e de programas
  - Diretórios
    - Organização e informações sobre arquivos

## Conceitos básicos

- Arquivos
  - Recipientes que contêm dados
- Diretórios
  - Conjuntos de referências a arquivos
- Partição
  - Abstração que permite - a partir do disco físico - criar discos lógicos
  - Cada partição possui sua estrutura própria de arquivos e diretórios
- Sistema de arquivos
  - Estrutura de como arquivos e diretórios estão organizados (*file system*)

## Controle de acesso e proteção

- Importante controlar o acesso a diretórios e arquivos devido a questões de segurança e de confidencialidade de dados
- Baseado na identificação dos usuários
  - Sistema de autenticação padrão (*login name* + *senha*)
  - Usuários possuem direitos de acessos
- Solução típica é o emprego de uma lista de acesso
  - Associar a cada arquivo e/ou diretório uma lista de acesso que determina que tipos de acessos são permitidos (leitura/escrita/execução) para diferentes classes de usuários (e.g.: proprietário, grupo e universo)

## UNIX: Acesso e proteção de arquivos

- O acesso a um arquivo pode ser de três tipos diferentes:
  - *r* : apenas leitura (*read*)
  - *w* : apenas escrita (*write*)
  - *x* : execução (*execute*)
- Combinados em bloco de três caracteres na ordem (leitura, escrita e execução)
  - Acesso negado é representado pelo caractere "traço" (-)
- Cada arquivo possui um bloco de proteção para três domínios diferentes: proprietário (*owner*), grupo (*group*) e outros (*others*)

## Acesso, proteção e tipos de arquivos em UNIX

- Acesso, proteção e tipo são fornecidos por uma sequência de 10 caracteres
- Tipo de arquivo:
  - arquivo normal (-)
  - Diretório (d)
  - link (l)
  - Dispositivo de E/S em modo bloco (b) ou caractere (c)

## Exemplo

```
drwxr-xr-x  2 asc  prof      4096 Aug 14 19:53 sag-0.6-port
-rw-r--r--  1 asc  prof      410287 Jul 20 15:43 securite1.pdf
-rw-r--r--  1 asc  prof      201854 Jul 20 15:43 securite2.pdf
-rw-rw-rw-  1 asc  prof      2101927 May  9 2000 seguranca.pdf
-rw-r--r--  1 asc  prof      548958 May  7 21:12 sendmail.pdf
-rw-r--r--  1 asc  prof      1369021 Jul 25 15:19 sisop-al.pdf
-rw-r--r--  1 asc  prof      1259776 Aug 15 19:08 slides.pdf
-rw-r--r--  1 asc  prof      278306 Jul 24 16:49 sockets.pdf
-rw-r--r--  1 asc  prof      16897 Aug 22 14:40 sumula.pdf
drwxr-xr-x  6 asc  prof      4096 Sep  3 11:19 tmp
-rw-r--r--  1 asc  prof      10274 Aug 15 10:02 tutorial_Linux.tgz
-rw-r--r--  1 asc  prof      148970 Apr 30 20:16 user.ps
-rw-r--r--  1 asc  prof      1543666 Jun 25 13:38 voice101.pdf
-rw-r--r--  1 asc  prof      120496 Mar 22 18:47 winbind.pdf
-rw-r--r--  1 asc  prof      30015 Jun  8 10:53 xcvlan.pdf
lrwxrwxrwx  1 asc  prof          7 Apr 30 20:16 ex -> user.ps
```

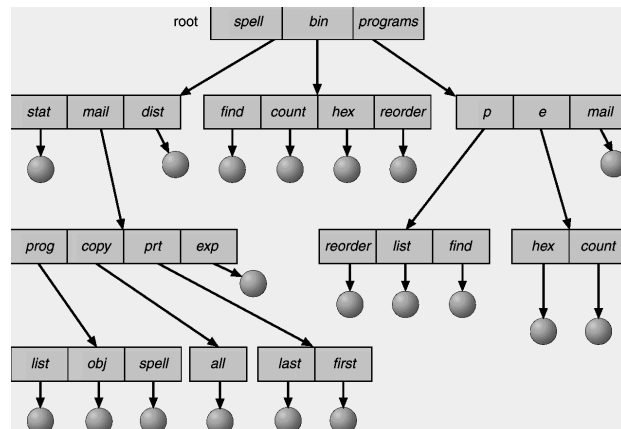
## Bits de proteção e acesso aplicados a diretórios

- Semântica dos bits de proteção é um pouco diferente se aplicada a diretórios
  - Acesso de leitura (*r*): permite verificar o conteúdo do diretório
  - Acesso de escrita (*w*): permite remover e inserir arquivos
  - Acesso de execução (*x*): permite o uso do comando *cd* (entrar no diretório)

## O conceito de diretório

- Estrutura de dados que contém informações sobre arquivos
  - Atributos
  - Localização
  - Propriedades
- Diretório é um arquivo do sistema operacional
  - Fornece um mapeamento entre o nome de um arquivo e o arquivo propriamente dito

## Diretório em árvore



Silberchatz, Galvin, Gagne. Applied Operating System Concepts (1st Ed.) John Wiley & Sons, 2000.

## Diretórios corrente, pai e raiz

- Diretório corrente (diretório de trabalho):
  - Qualquer folha da árvore
  - Representado por "."
- Diretório pai:
  - Representado por ".."
- Diretório raiz:
  - Representado por "/"

## Caminho absoluto e relativo

- Qualquer arquivo (ou subdiretório) pode ser identificado de forma não ambígua através de seu caminho (*pathname*)
- Caminho absoluto
  - Quando se referencia um arquivo a partir da raiz da árvore
    - e.g.: */spell/mail/prt/first*
- Caminho relativo
  - Quando se referencia um arquivo a partir do diretório corrente
    - e.g.: *prt/first*

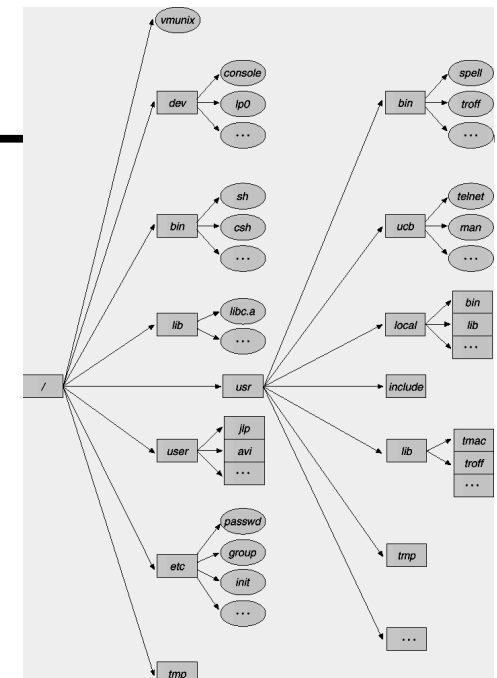
## Diretório estruturado em grafo

- Generalização da estrutura em árvore que surge como solução para compartilhamento de arquivos
- Compartilhamento pode ser implementado através de aliases
- *Link* é uma forma comum de alias (sinônimo)
  - Ponteiro para outro arquivo ou subdiretório
  - Link simbólico
- *Link* é uma entrada na estrutura de diretório
  - Possui o caminho para o arquivo propriamente dito
- Remover um *link* implica em remover apenas a sua entrada na estrutura de diretório, não o arquivo que aponta

## Organização de diretórios do UNIX

- Baseado em partições
- Diretório raiz do sistema de arquivos corresponde a uma partição especial (*root*)
- Pontos importantes:
  - Cada partição possui seu próprio sistema de arquivos
  - Capacidade de integrar diferentes sistemas de arquivos em uma mesma hierarquia
    - Conceito de ponto de montagem
  - Sistemas de arquivos podem ser diferentes
    - e.g.: ext2, FAT12, FAT32, NTFS, etc...

## Organização típica dos diretórios UNIX



## O conceito de partição

- Particionar significa dividir um disco físico em vários discos lógicos
- Porque particionar?
  - Encapsular dados
    - Problemas no sistema de arquivos são locais a uma partição
  - Aumentar a eficiência do disco:
    - O tamanho do bloco é um atributo da partição
  - Limitar demanda de crescimento de espaço em disco para uma partição
    - e.g.; `/var/messages`

## Organização interna de uma partição

- Uma partição é um disco lógico
- Cada partição é autocontida, isto é, todas as informações para acesso aos arquivos da partição estão contidas na própria partição
  - Diretórios e subdiretórios
  - Descritores de arquivos da partição
  - Blocos de dados
  - Lista de blocos livres da partição
- Formatação lógica corresponde a inicialização dessas estruturas de dados
- Normalmente um setor (bloco) especial do disco informa quais são as partições e quais parcelas do disco a partição ocupa

## Hard links e soft links

- Em sistemas UNIX o conceito de *aliases* é implementado através de *hard links* e *soft links* (simbólicos)
- Diferenças entre *hard links* e *soft links*
  - Só é possível criar um *hard link* dentro de um mesmo sistema de arquivos
  - É possível facilmente identificar um *soft link* ao passo um *hard link* não
- Exemplos:

-rw-r--r--	2	mary	staff	25340	Aug 12 16:51	bar
lwxrwxrwx	1	mary	staff	1024	Aug 12 17:01	foo -> winbind.pdf
-rwxr--r--	1	mary	staff	214056	May 30 22:19	windbind.pdf

## Roteiro

- Conceitos Básicos de Sistemas Operacionais
- Gerência de Processos
- Sistemas de Arquivos
- Sistemas de Entrada e Saída
- Comandos UNIX "hiper-básicos"



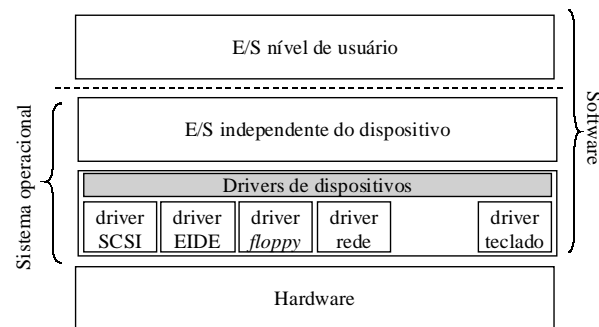
## Objetivos da gerência de entrada e saída

- Eficiência
- Generacidade é importante
  - Desejável que dispositivos sejam tratados da forma mais uniforme possível
- Esconder os detalhes do serviço de entrada e saída em camadas de mais baixo nível
- Fornecer ao alto nível abstrações genéricas como *read*, *write*, *open* e *close*
- Envolve aspectos de hardware e de software

## Princípios básicos de software de entrada e saída

- Subsistema de entrada e saída é software bastante complexo devido a diversidade de periféricos
- Objetivo é padronizar as rotinas de acesso aos periféricos de E/S de forma a reduzir o número de rotinas
  - Permite inclusão de novos dispositivos sem alterar "visão" do usuário (interface de utilização)
- Para atingir esse objetivo o subsistema de E/S é organizado em camadas

## Estrutura em camadas do subsistema de E/S



## Organização lógica do software de E/S

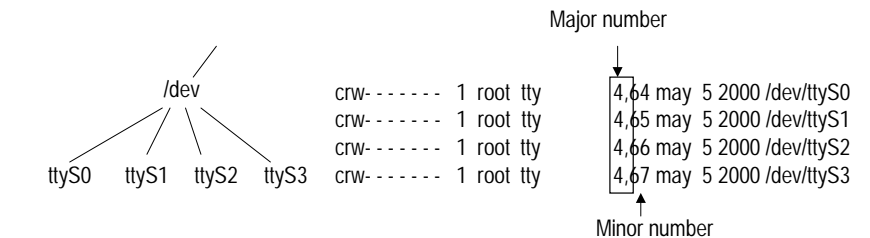
- Segue uma filosofia modular (três módulos base)
  - Dispositivo lógico de E/S
    - Trata dispositivo como recurso lógico sem se preocupar com detalhes de controle
    - Oferece uma interface de programação ao processo usuário
  - Dispositivo físico de E/S
    - Aceita solicitações abstratas e converte para um tratamento específico
    - Bufferização
  - Escalonamento e controle
    - Tratamento de interrupções e de status
- Organização não é única
  - Depende do tipo do dispositivo e de sua aplicação

## Drivers de dispositivos

- Camada inferior do software de entrada e saída
- Responsável pela gerência do dispositivo físico de E/S
- Código específico a um dispositivo
- Recebe solicitações da camada de gerência de dispositivo lógico
  - Exporta uma “visão” uniforme subdividindo os periféricos de E/S em função da unidade de transferência de dados:
    - Orientados a bloco
    - Orientados a caracteres
    - Pseudo-dispositivos
- Responsável pelo tratamento de erros

## Entrada e saída independente do dispositivo

- Fornece uma visão lógica do dispositivo através de estruturas de dados genéricas que representam classes de dispositivos
- Atribuição uniforme do nome independente do dispositivo
- O UNIX é um exemplo clássico:
  - Nome do dispositivo é um string
  - Discos (dispositivos) fazem parte do sistema de arquivos



## Entrada e saída a nível de usuário

- “Visão” que o usuário possui dos dispositivos de entrada e saída
  - Interface de programação associada a bibliotecas de entrada e saída
  - Interface de “comandos” (`ls`, `cp`, `mv`, etc)
- Construído sobre primitivas do sistema operacional (chamadas de sistema)

## Disco magnético

- Mais importante dispositivo de entrada e saída
- Formatação física consiste em criar trilhas e setores
  - Feito na fábrica
- Formatação lógica prepara o disco para armazenar e recuperar arquivos
  - Consiste em criar um sistema de arquivos
  - Particionamento

## Conceito de partição

- Uma partição é um disco lógico
- Particionar significa dividir um disco físico em vários discos lógicos
- Uma partição é auto-contida, isto é, ela é um sistema de arquivos independente
  - Estrutura própria de diretórios e subdiretórios
  - Tamanho de bloco
  - Gerência de espaço livre e ocupado
  - etc
- Um setor (bloco) especial do disco informa quais são as partições e quais parcelas do disco a partição ocupa

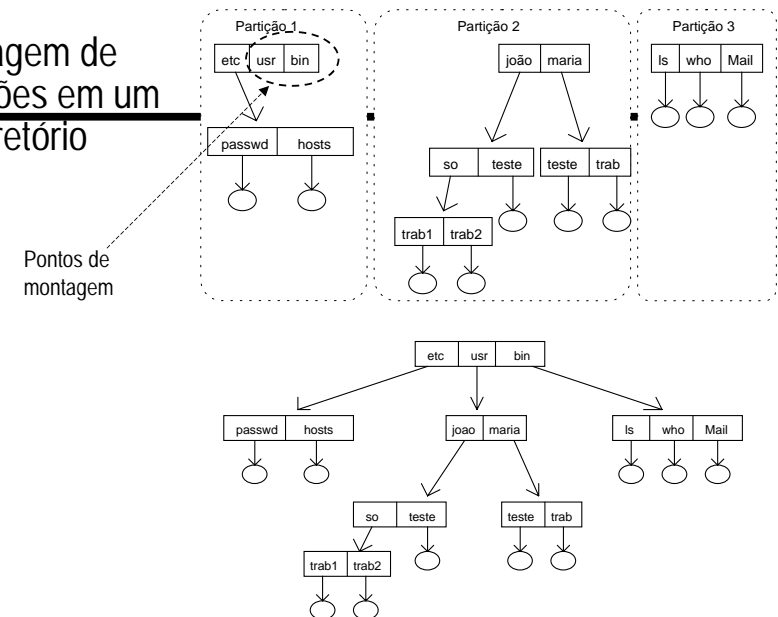
## Partições típicas

- Partição do sistema
  - Estrutura de diretórios e arquivos do sistema operacional
- Partição de *boot*
  - Onde está o sistema operacional a ser carregado
- Partição de dados
  - Estrutura de diretórios e arquivos
- Partição de swap
  - Associada ao mecanismo de gerência memória (memória virtual)

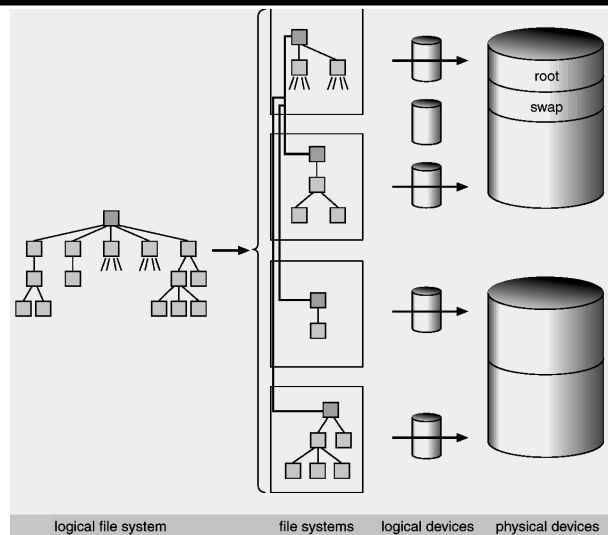
## O conceito de montagem

- Meios de armazenamento necessitam ter um sistema de arquivos
- Individualizado por meio de armazenamento
  - e.g.: Disquetes, cd-rom, fitas, etc, cada um possui o seu próprio sistema de arquivos
- Montar um sistema de arquivo significa integrá-lo a um outro sistema de arquivos já existente
  - Ponto de montagem
- Sistema de arquivos pode estar em outra máquina (montagem remota)
  - e.g.: Network File System (NFS)
    - Conceito de exportação e de importação

## Montagem de partições em um subdiretório



## Exemplo de montagem



## Roteiro

- Conceitos Básicos de Sistemas Operacionais
- Gerência de Processos
- Sistemas de Arquivos
- Sistemas de Entrada e Saída
- Comandos UNIX "hiper-básicos"

## Comandos básicos associados a processos

- Manipulação de processos
  - `ps` : lista os processos em execução
    - e.g.: `ps -aux`
  - `kill` : remove um processo
    - e.g.: `kill -9 918`
  - `top` : lista de processos ativos e informações sobre o sistema
- Composição de comandos
  - Pipe (`|`) permite de se "ligar" comandos UNIX entre si fazendo com que a saída de um comando seja a entrada de outro
    - e.g.: `ps -aux | grep root`

## Comandos básicos associados a arquivos (1)

- Posiciona o diretório corrente: `cd path`
  - Pode ser a partir do diretório corrente ou absoluto
  - Exemplos:
    - `cd capitulo`
    - `cd capitulo/amostras`
    - `cd /home/asc/public_html`
- Criar/remover diretórios: `mkdir / rmdir`
  - Exemplos:
    - `mkdir capitulo`
    - `mkdir -f capitulo/amostras`
    - `rmdir capitulo`

## Comandos básicos associados a arquivos (2)

### ■ Listar o conteúdo de um diretório: ls

#### ■ Exemplos:

```
ls -la *  
ls -la *.pdf  
ls arquivo?
```

### ■ Copiar arquivos: cp

#### ■ Exemplos:

```
cp arquivo1 arquivo2  
cp /etc/shell .
```

### ■ Remover arquivos: rm

#### ■ Exemplos:

```
rm arquivo1
```

## Comandos básicos associados a arquivos (3)

### ■ Criar links: ln

#### ■ Exemplos:

```
ln teste texto  
ln -s foo windbind.pdf
```

### ■ Mover arquivos: mv

#### ■ Exemplos:

```
mv teste texto  
mv arquivo1 ../public_html/.
```

### ■ Verificar conteúdo de arquivos: more e cat

#### ■ Exemplos:

```
more arquivo1  
cat arquivo
```

## Comandos básicos associados a arquivos (4)

### ■ Modificar os direitos de acesso:

#### ■ `chmod quem-modificação-permissão arquivo`

- quem: u (user) g (group) o (others)
- modificação: + (atribuir) - (remover)
- permissão: *rwx*
- Exemplo:

```
chmod u+rwx fonte.c
```

#### ■ Atribuição através da codificação numérica dos blocos de sequência

```
chmod 644 texto.txt
```

## Comandos básicos associados a arquivos (5)

### ■ Modificar o proprietário: `chown proprietário arquivo`

#### ■ Exemplo:

```
chown bill dados
```

### ■ Modificar o grupo: `chgrp grupo arquivo`

#### ■ Exemplo:

```
chgrp projeto dados
```

## Comandos básicos associados a sistemas de arquivos

---

### ■ Montar um sistema de arquivo: mount

#### ■ Exemplos:

```
mount -t iso9660 /dev/hdb /mnt/cdrom  
mount /dev/cdrom /mnt/cdrom
```

### ■ Desmontar um sistema de arquivo: umount

#### ■ Exemplos:

```
umount /mnt/cdrom
```

### ■ Criar um sistema de arquivos: mkfs

#### ■ Exemplos:

```
mkfs -t ext2 /dev/hda1
```

## Aprendendo a “se virar” no mundo UNIX...

---

### ■ Os comandos UNIX (Linux) são bastante flexíveis oferecendo uma série de opções de execução

### ■ Comando (talvez) mais importante a aprender a usar é o man

#### ■ `man` : fornece documentação de uso de um comando UNIX qualquer

#### ■ Exemplo:

```
man ls
```

### ■ Existem outros mecanismos (info, apropos, etc...)

#### ■ No Linux existem HOWTOs

#### ■ Na instalação de um sistema pode-se optar por não disponibilizar a documentação (não recomendável, ao menos para o man)