FACULDADE DE CIÊNCIAS E TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA

Fundamentos de Sistemas de Operação

LEI - 2023/2024

Vitor Duarte
Mª. Cecília Gomes

1

Aula 16

- Dispositivos de armazenamento
- Gestão de Ficheiros
  - Implementação do SF

- OSTEP: cap. 39, 40

- revisão AC: OSSTEP cap. 36, 37, 38, 44

2

Gestão de ficheiros e discos

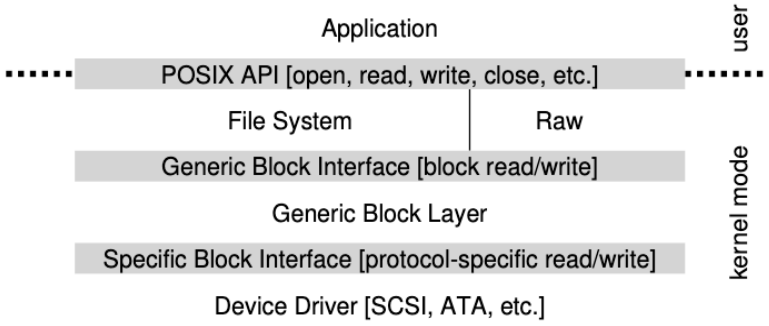


Figure 36.4: **The File System Stack**

3

Arquitetura de IO (exemplo)

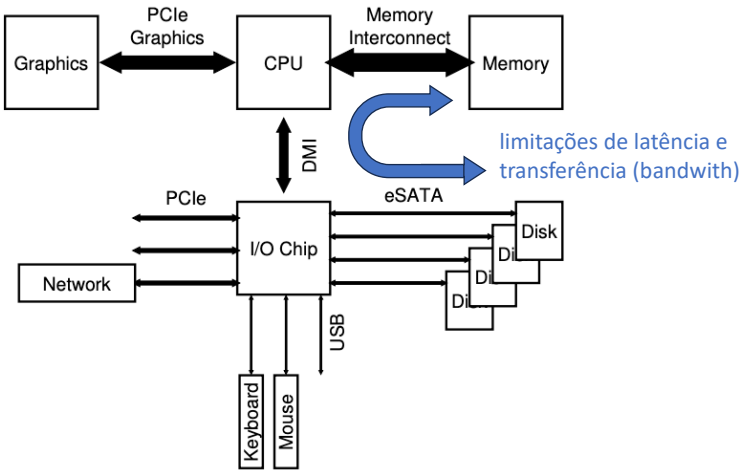


Figure 36.2: **Modern System Architecture**

4

Algumas características dos "discos"

- Muito dependente de vários fatores:
  - BUSes e controladores IO
  - controlador interno ao disco
  - caracteristicas das memórias (SSD) e do disco (p.e. RPM)

	Random		Sequential	
Device	Reads (MB/s)	Writes (MB/s)	Reads (MB/s)	Writes (MB/s)
Samsung 840 Pro SSD	103	287	421	384
Seagate 600 SSD	84	252	424	374
Intel SSD 335 SSD	39	222	344	354
Seagate Savvio 15K.3 HDD	2	2	223	223

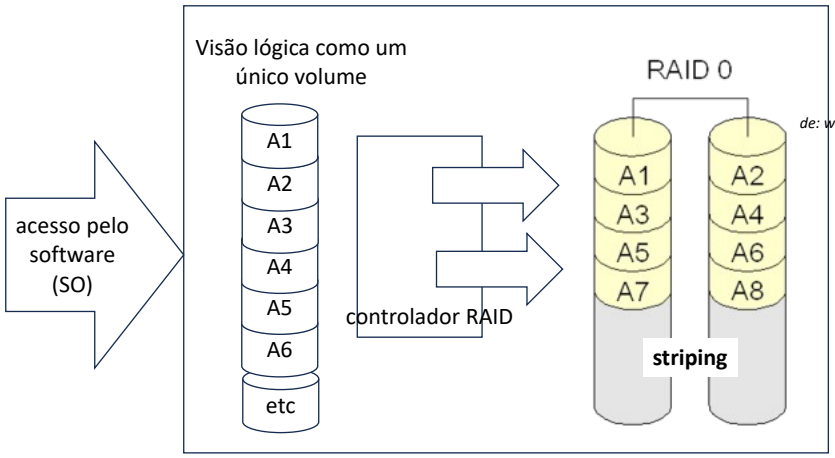
Figure 44.4: **SSDs And Hard Drives: Performance Comparison**

- Conclusões mais relevantes:
  - Melhor desempenho em SSD
  - A ordem dos acessos tem impacto no desempenho
    - Nos HDD depende muito dos *seek* e *rotation* necessários
  - Melhores desempenhos para pedidos de grandes volumes de dados contíguos

Agregando discos num volume

- RAID - *Redundant Array of Inexpensive/Independent Disks*
- Objetivos:
  - Redundância - tolerar falhas dos discos
  - Desempenho - latência e taxa de transferência
- Como:
  - Replicando os dados - cópias ou códigos de correção
  - operando em paralelo - sobre vários discos em simultâneo
- Várias organizações dos discos: RAID 0, 1, ...,6
- Idealmente implementado nos controladores ou em *disk arrays*
  - se por software (driver no SO) tem fraco desempenho

Exemplo: RAID 0

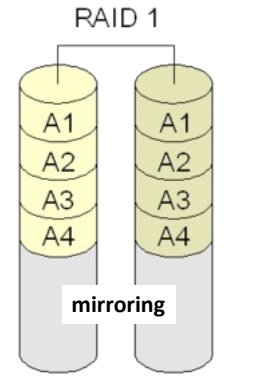


capacidade é a soma dos volumes físicos

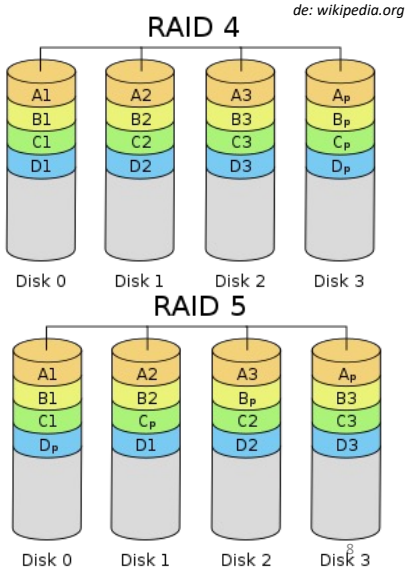
striping dá desempenho mas não tem replicação

7

Mais Exemplos



mirroring dá replicação mas não melhora desempenho nas escritas



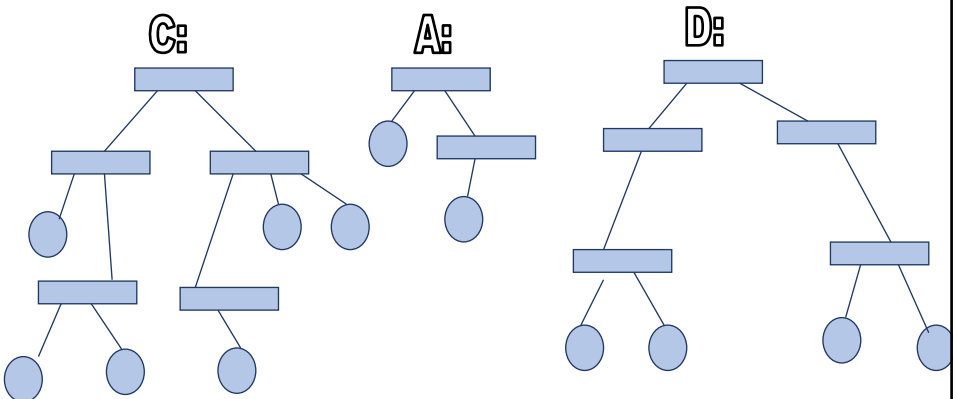
8

Volumes

- Para o SO existem Dispositivos (*devices*) que são Volumes de dados onde guardar informação
  - Um volume pode ser um disco ou parte de um disco: partição
  - Um volume pode esconder uma agregação de discos em RAID
  - Um volume pode ser uma parte da RAM
- Gestão de ficheiros usa uma interface genérica para ler e escrever blocos nos volumes, suportada pelos *device-drivers*
  - (um bloco = vários setores contíguos)
- Para guardar ficheiros no disco, o SO tem de gravar nele as suas próprias estruturas de dados
  - Formatar: Criar nesses blocos, num formato pré-definido, as estruturas de dados necessárias → "o sistema de ficheiros em disco"
  - Ou formatação para swap/paginação.
  - **`format/diskpart`** (windows), **`mkfs`** (unix/linux)
- Um disco pode começar por um Boot block, código de arranque do SO (os 1ºs sectores)

Vários discos lógicos separados

- Cada volume tem o seu Sistema de Ficheiros
- Em MS-Windows, normalmente, cada um corresponde a uma "drive"

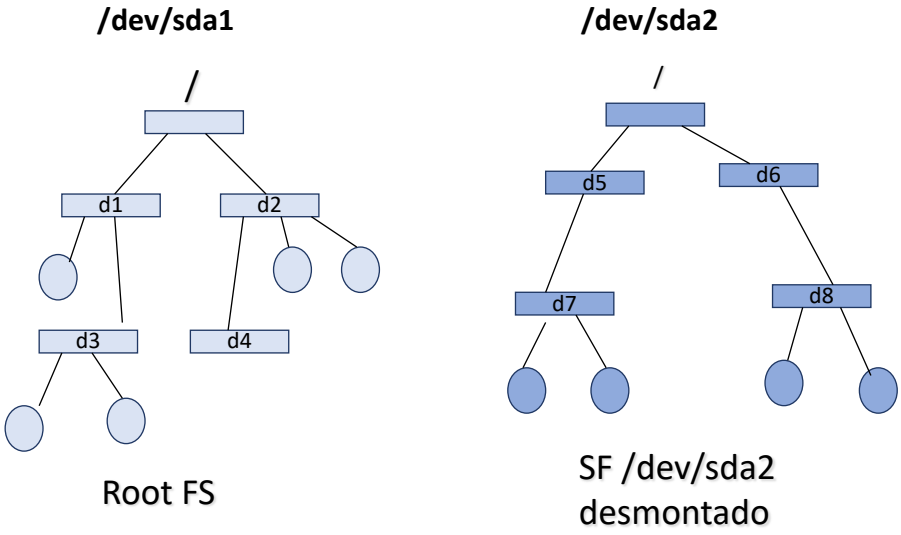


Sistema de ficheiros integrado

- Em Unix, há um sistema de ficheiros integrado
- O *root file system* está presente desde o início (*boot*)
- Qualquer outro volume tem de ser integrado na hierarquia já existente, para ser acessível aos processos
- O SF no volume é colocado (mounted) numa diretoria que funciona como mount point.
- É possível que cada volume tenha um sistema de ficheiros diferente

11

SF corrente /dev/sda1

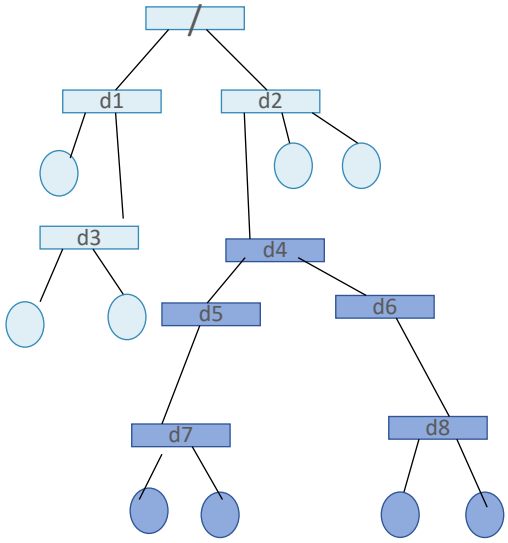


Root FS

SF /dev/sda2 desmontado

12

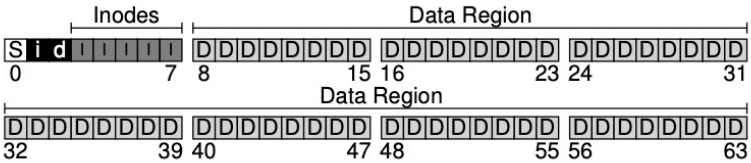
Após `mount /dev/sda2 /d2/d4`



13

UNIX file systems

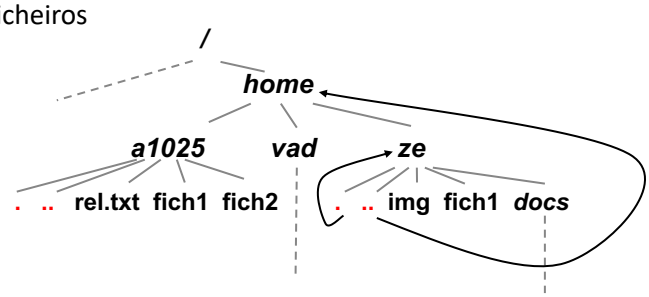
- Os SF no Unix incluem normalmente:
  - Superblock: descrição do volume e da formatação
    - id do SF, sua dimensão, nº de inodes e de blocos de dados, dim dos blocos, root inode, etc.
  - bitmap de inodes e bitmap de blocos de data: que inodes e blocos de dados estão livres ou em uso
  - Os inodes: informação de todos os nós (ficheiros, diretorias, etc),
  - Blocos de Dados: os blocos com o conteúdo das diretorias e dos ficheiros



14

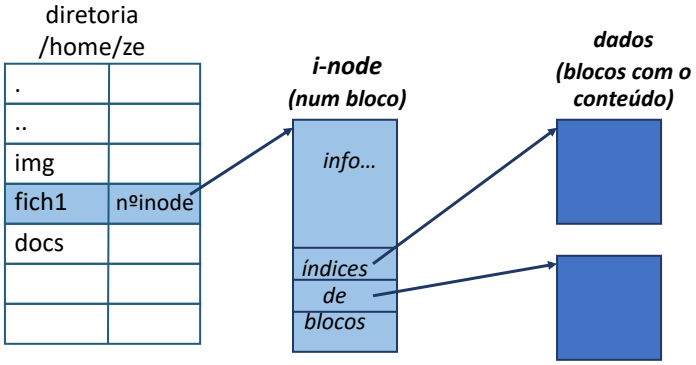
Visão do utilizador: espaço de nomes

- Espaço de nomes hierárquico
- Cada nome (nó) pode ser uma diretoria, um ficheiro em disco ou noutro dispositivo...
  - Cada diretoria contém uma lista de outras diretorias e/ou ficheiros



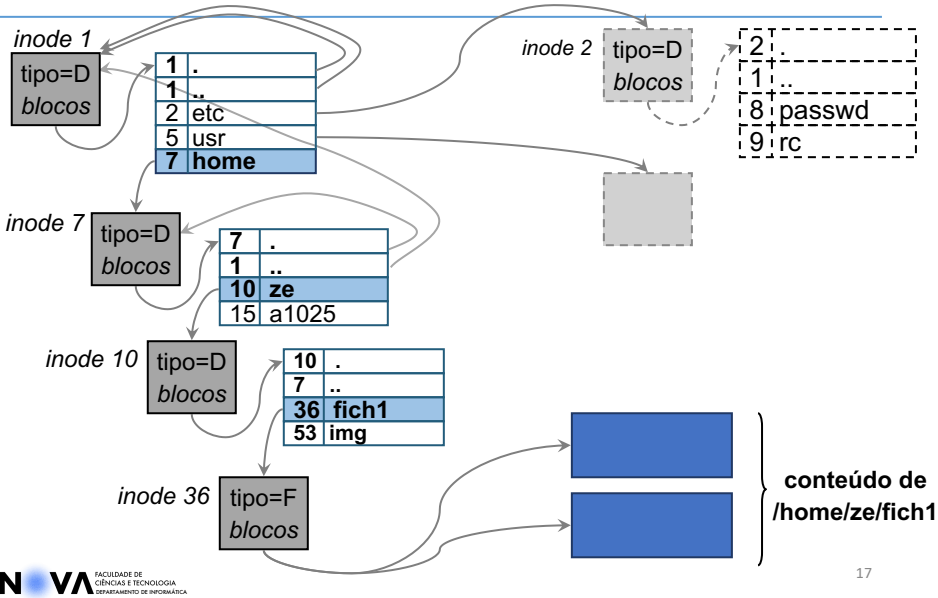
Estrutura de diretorias e ficheiros

- **Um** ficheiro/diretoria = **um** i-node e **pelo menos um** nome



- O par (id.volume, i-node) identifica univocamente estes "objectos" no sistema

Caminho /home/ze/fich1



17

Obtendo informação (stat)

```
int stat(char *path, struct stat *info)
int fstat(int fd, struct stat *info)
```

Coloca em *info* informação obtida tipicamente do *inode* :

```
struct stat {
    dev_t  st_dev;  /* ID of device */
    ino_t  st_ino;  /* inode number */
    mode_t st_mode; /* protection and type */
    uid_t  st_uid;  /* user ID of owner */
    off_t  st_size; /* total size, in bytes */
    time_t st_mtime;/* time of last modific.*/
    nlink_t st_nlink;/* number of hard links */
    . . .
};
```


18

Nomes e Identificadores

- Nome de ficheiro pode ser quase qualquer coisa (não pode incluir "/")
  - associado nas diretorias ao nº inode do ficheiro
- Cada utilizador tem um User ID (UID) e pertence a um grupo de utilizadores (GID)
  - Têm nomes associados nas BDs de utilizadores e grupos
  - Cada processo herda o UID e GID do processo pai
- Cada processo tem também *Efective UID* e *Efective GID* que são normalmente iguais aos UID e GID
  - Podem ser diferentes se o programa executado tiver permissão para executar como outro utilizador (*set uid* ou *set gid*)
  - EUID e EGID são usados para validar o acesso aos ficheiros.
- Chamadas ao sistema que obtém os identificadores para o processo:

```
int getuid(void)/ int geteuid(void)
int getgid(void)/ int getegid(void)
```

- O "superuser" (root – uid 0) tem normalmente todos os privilégios.

Permissões de acesso a ficheiros

- Cada nó do SF tem tem o uid do dono (‘owner’) e do grupo (`group´)
  - do processo que os criou ou explicitamente alterado por **`chmod`**
- Cada nó do SF define as permissões por classes de utilizadores :
  - owner: cujo UID foi associado ao ficheiro na sua criação ou explicitamente indicado
  - group: grupo (GID) a cujo UID pertence ou explicitamente indicado para o ficheiro
  - other: nem dono nem grupo
- Permissões nos modos de acesso (a quando de open ou execve):
  - ler (Read) – escrever (Write) – executar (eXecute)
- Exemplo (várias representações):

```
 owner   group   other
  rwx     r-x     r--      (texto: ls -l)
  111     101     100      (binário)
   7       5       4       (base octal)
S_IRWXU            S_IROTH  (defines em sys/stat.h)
     S_IRGRP|S_IXGRP
```

Máscara de criação de ficheiros

`mode_t umask(mode_t newmask)`

- Cada processo tem um 'umask' associado
- Sempre que cria um ficheiro ou uma diretoria, os bits que estiverem a 1 em 'umask' forçam a 0 os bits correspondentes no *mode* das permissões
- Exemplo: `oldmask = umask(022);`

  define: 000 010 010
  --- -w- -w-

  `fd = creat("f", 0666);`
  - em vez ficar: 110 110 110 (0666 = rw-rw-rw-)
  - fica: 110 100 100 (0644 = rw-r--r--)

Nomes: link e unlink

- Podemos ter mais nomes para o mesmo ficheiro
  - 1º nome com **`open/create`**
  - acrescentar nome a diretoria referindo-se ao mesmo inode (não podem ser diretorias)

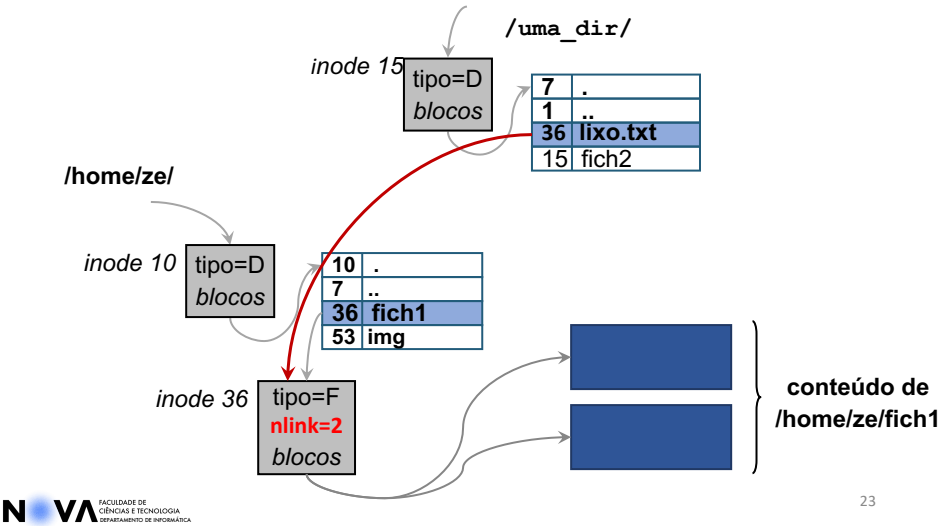
    `int link(char *name, char *newname)`
  - links simbólicos ou atalhos (não conta como link)

    `int symlink(char *name, char *newname)`
- Um ficheiro é **apagado** quando não tem mais nomes (*links*)

  `int unlink(char *name)`
- Para diretorias: **`mkdir`** e **`rmdir`**
  - chamadas complicadas… O que o SO precisa fazer?
- Na linha de comados temos: **`ln, rm, mkdir, rmdir`**

links (hard)

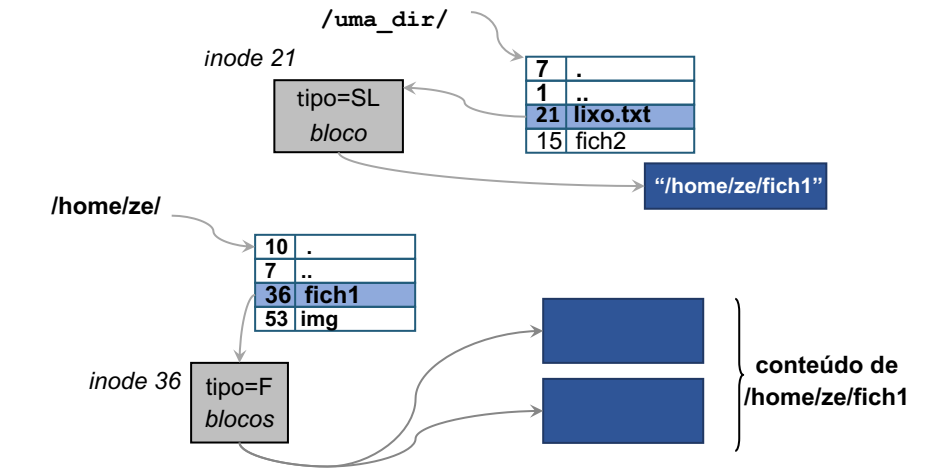
```
ln /home/ze/fich1 /uma_dir/lixo.txt
```



23

Links simbólicos

```
ln -s /home/ze/fich1 /uma_dir/lixo.txt
```



24

Renomear/mover: rename

```
int rename(char *old, char *new)
```

- Muda o nome de ficheiros e de diretorias
  - permite mudar o nome e mover de diretoria
- Para as diretorias mantém a sua consistência
  - (. e ..)
- garante que o ficheiro não desaparece
  - Com link e unlink poderiamos ter o mesmo efeito mas sujeitos a perder o ficheiro ou duplicar