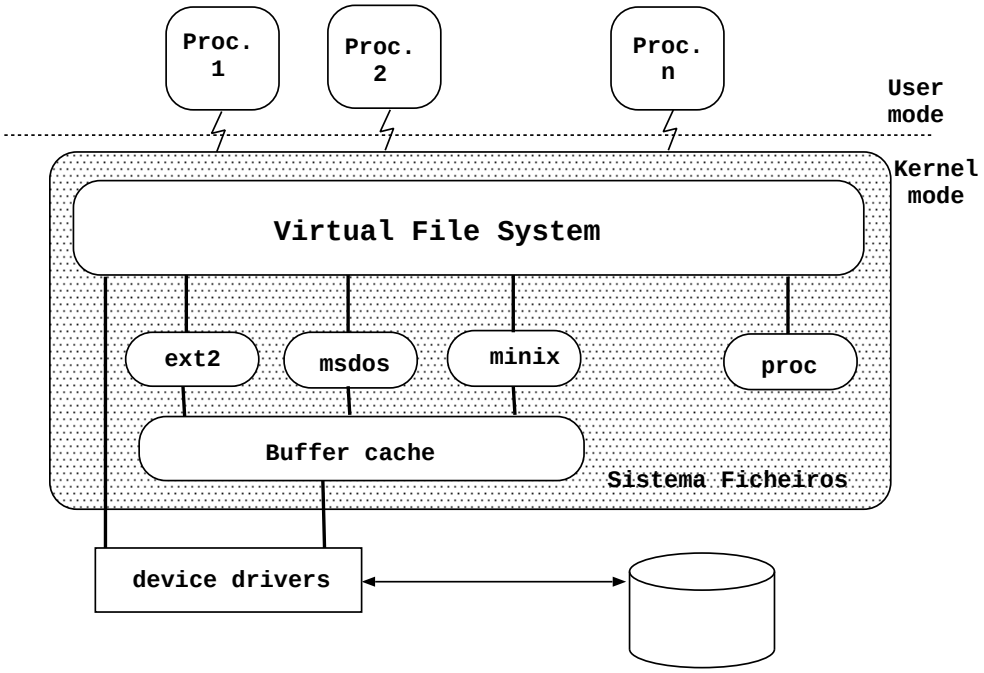
Linux: sistema de ficheiros virtual

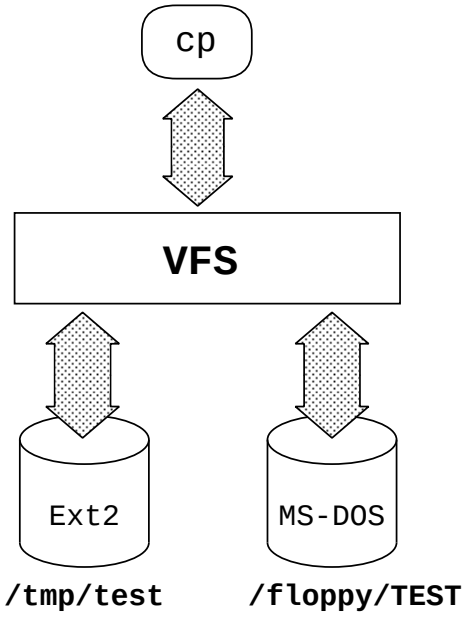
- O SF do Linux aparenta ser uma hierarquia de directórios organizada em árvore semelhante aos restantes sistemas UNIX.
- Mas internamente, o kernel consegue gerir múltiplos sistemas de ficheiros diferentes através de uma camada abstracta unificadora, o Virtual File System (VFS).

  Esta é uma das razões porque o Linux se tornou tão popular.



VFS: um exemplo

Considere que executa o seguinte comando `cp /floppy/TEST /tmp/test` em que `/floppy` está montado numa disquete MS-DOS e `/tmp` é um directório normal Ext2.




```
inf= open("/floppy/TEST", O_RDONLY, 0);
outf= open("/tmp/test",
       O_WRONLY|O_CREATE|O_TRUNC, 0600);
do {
  st= read(inf,buf,4096);
  write(outf,buf,st);
} while (st);
close(outf);
close(inf);
```

Neste caso, `cp` interage com o VFS através de chamadas de sistema genéricas, sendo responsabilidade do VFS lidar com os diferentes formatos de sistemas de ficheiros.

O que acontece realmente:

`cp` invoca `read()` do VFS; O kernel traduz a chamada `read()` em `sys_read()`;

A estrutura de dados que representa o ficheiro em memória inclui um campo, `f_op`, que inclui um apontador para funções especificas de um SF. Assim, a chamada passa para:

```
file->f_op->read(...);
```

que é específica do MS-DOS.

VFS e Ext2fs

- O VFS do Linux está concebido de acordo com os princípios OO e é constituído por duas componentes:
  - Um conjunto de definições que caracterizam um objecto do tipo ficheiro:
    - blocos-inode e blocos-dados representam os ficheiros
    - um objecto file-system representa todo o sistema de ficheiros
  - Uma camada de software que manipula esses objectos
- O Ext2fs é muito semelhante ao BSD Fast File System (ffs), na estrutura (bloco de boot, superbloco, blocos de inodes e blocos de dados) e na forma de localizar blocos de dados de um dado ficheiro (inodes).

  Ultrapassa a limitação de 2Gbytes existentes no sistema de ficheiros `ext`. Inicialmente, o Linux baseava-se em `minix` e tinha uma limitação de 64Mbytes para tamanho de uma partição.
- As principais diferenças entre `ext2fs` e `ffs` estão na atribuição de espaço em disco:
  - `ffs` usa blocos de 8Kb que podem ser divididos em fragmentos de 1Kb.
  - `ext2fs` não usa fragmentos, usa blocos menores de 1Kb (default) ou 2Kb.

Principais caracteristicas do Ext2fs

- suporta os ficheiros standard em UNIX: ficheiros regulares, directórios, ficheiros especiais de periféricos, e links simbólicos.
- permite nomes de ficheiros muito grandes (255 caracteres e pode passar para 1012 se necessário). Usa entradas de directório de comprimento variável.
- reserva alguns blocos para a `root`. Normalmente 5% dos blocos. A ideia é reter espaço para recuperação caso um processo encha o sistema de ficheiros.
- suporta algumas extensões que não estão presentes em boa parte dos SOs Unix:
  - fixar o tamanho lógico de um bloco
  - forma de sincronizar a actualização dos blocos em disco
  - semântica de criação de ficheiros: herdam group-id do directório pai (BSD) ou depende do bit setgid do directório-pai (Sys V).
  - traçar o estado do sistema de forma a que quando no arranque caso seja necessário se proceda a uma verficação de consistência do sistema.
- **Ext3fs**: análogo ao Ext2fs mas suporta journaling.

O sistema de ficheiros Proc

- providencia informação sobre o estado corrente do kernel do Linux e dos processos em execução.
- não guarda dados, em vez disso o seu conteúdo é calculado dinamicamente de acordo com os pedidos de I/O feitos pelo utilizador.
- é montado normalmente em `/proc`.
- para cada processo em execução associa um directório, `/proc/pid`, em que `pid` é o identificador do processo.

  Este directório contém ficheiros que caracterizam o estado do processo.
- muitos dos comandos Linux baseiam-se neste sistema de ficheiros, e.g. o comando `ps`.

Directórios e ficheiros do Proc

- **loadavg**: ficheiro com os valores de carga média do sistema nos últimos 1, 5 e 15 minutos.

  Fornece ainda o número de processos em execução, o número total de processos e o PID do último processo activo.

  Exemplo: `0.00 0.11 0.08 1/97 2898`

- **uptime**: ficheiro com o tempo em segundos desde que o sistema arrancou.

- **meminfo** ficheiro que descreve o estado da memória principal (total, livre e ocupado). Fornece ainda informação sobre o total de memória partilhada por processos, assim como informação sobre a cache.

  Análoga ao comando `free`, só que está em bytes.

```
        total:    used:    free:  shared: buffers:  cached:
Mem:  261824512 253251584  8572928       0 40206336 62697472
Swap: 263200768  6791168 256409600
MemTotal:       255688 kB
MemFree:          8372 kB
MemShared:           0 kB
Buffers:         39264 kB
Cached:          59288 kB
SwapCached:       1940 kB
Active:          88132 kB
Inactive:        82072 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:       255688 kB
LowFree:          8372 kB
SwapTotal:      257032 kB
SwapFree:       250400 kB
NrSwapPages:     62600 pages
```

Directórios e ficheiros do Proc (cont.)

- **version** ficheiro com a informação da versão do Linux corrente (variável `linux_banner`:

```
Linux version 2.4.18-6mdk (quintela@bi.mandrakesoft.com)
(gcc version 2.96 20000731 (Mandrake Linux 8.2 2.96-0.76mdk))
#1 Fri Mar 15 02:59:08 CET 2002
```

- **cpuinfo** ficheiro com os parâmetros do processador identificados durante a inicialização (boot) da máquina:

```
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 8
model name      : Pentium III (Coppermine)
stepping        : 10
cpu MHz         : 701.596
cache size      : 256 KB
fdiv_bug        : no
hlt_bug         : no
f00f_bug        : no
coma_bug        : no
fpu             : yes
fpu_exception   : yes
cpuid level     : 2
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 sep mtrr ...
bogomips        : 1399.19
```

- **pci** ficheiro com informação sobre os encaixes PCI.

```
PCI devices found:
  Bus  0, device   0, function  0:
    Host bridge: Intel Corp. 440BX/ZX - 82443BX/ZX Host bridge (rev 3).
      Master Capable.  Latency=32.
      Prefetchable 32 bit memory at 0xf4000000 [0xf7ffffff].
  Bus  0, device   1, function  0:
    PCI bridge: Intel Corp. 440BX/ZX - 82443BX/ZX AGP bridge (rev 3).
      Master Capable.  Latency=32.  Min Gnt=140.
...
```

Directórios e ficheiros do Proc (cont.)

- **net/** directório com os ficheiros que descrevem a camada de rede do Linux.
- **malloc** monitoriza as operações `kmalloc()` e `kfree()`.
- **kcore** "core dump" do kernel, útil para debugging.
- **modules** módulos carregados, o seu tamanho e estado.
- **stat** estatísticas gerais do kernel.
- **devices** periféricos registados, nomeadamente o número que lhe está associado.
- **interrupts** números e nomes de interrupts de hardware recebidos.
- **filesystems** implementações de SFs existentes no kernel.

```
nodev   rootfs
nodev   proc
        ext2
nodev   devfs
nodev   devpts
        reiserfs
nodev   supermount
        iso9660
        vfat
```

- **mounts** lista dos SFs presentemente montados.

```
/dev/root / reiserfs rw,noatime 0 0
none /dev devfs rw 0 0
none /proc proc rw 0 0
none /dev/pts devpts rw 0 0
none /dev/shm tmpfs rw 0 0
/dev/hda4 /home reiserfs rw,noatime 0 0
/mnt/cdrom /mnt/cdrom supermount ro 0 0
/mnt/floppy /mnt/floppy supermount ro,sync 0 0
none /tmp tmpfs rw 0 0
none /proc/bus/usb usbdevfs rw 0 0
```

O directório /proc/net/

Contém ficheiros que descrevem o estado da camada de rede:

- **unix** info sobre cada socket UNIX aberto (caminho, estado, tipo, flags, protocolo e contador de referências):

```
Num       RefCount Protocol Flags    Type St Inode Path
cff02a60: 00000010 00000000 00000000 0002 01  2829 /dev/log
c27a45c0: 00000003 00000000 00000000 0001 03 10305
```

- **arp** conteúdo da tabela arp (address resolution protocol – converte endereços IP em em endereços reais no hardware).
- **route** tabela de rotas num formato pouco comum. O comando `route` usa esta informação.

```
Iface Destin.  Gateway Flags RefC Use Metric   Mask   MTU Window IRTT
eth0 0000000A 00000000 0001    0   0    0    00FFFFFF 40    0     0
```

  "`route -n`" dá informação mais legível.

- **dev** periféricos de rede disponíveis e info estatística do seu uso.

```
Inter-| Receive                          | Transmit
face  | bytes   packets errs drop ... | bytes  packets errs drop ...
    lo: 1123222   11467    0    0 ...  1123222   11467    0    0 ...
  eth0:       0       0    0    0 ...     9960     166    0    0 ...
```

- **sockets** estatísticas sobre sockets

O directório /proc/sys/

Contém subdirectórios com info relevante sobre o sistema:

- **kernel** info sobre o kernel e estruturas de control;
- **fs** info sobre parâmetros do SF
  - **file-max** número máximo permitido de ficheiros abertos
  - **file-nr** número de ficheiros abertos
  - **inode-max** número total de inodes
  - **inode-nr** número de inodes usados
- **vm** info sobre parâmetros de memória virtual
- outros directórios: **net**, **dev**, etc.

Info sobre processos

Cada processo tem um directório associado com ficheiros com info relevante:

- **status** características do processo: estado, pid, ppid, uid, gid, vmsize ...

```
Name:   ntpd
State:  S (sleeping)
Tgid:   1039
Pid:    1039
PPid:   1
TracerPid:      0
Uid:    0       0       0       0
Gid:    0       0       0       0
FDSize: 32
Groups:
VmSize:     1908 kB
VmLck:      1908 kB
VmRSS:      1900 kB
VmData:      124 kB
VmStk:        20 kB
VmExe:       200 kB
VmLib:      1516 kB
SigPnd: 0000000000000000
SigBlk: 0000000000000000
SigIgn: 8000000000001000
SigCgt: 0000000000016a47
```

- **root** link para o directório root
- **exe** link para o executável do processo
- **fd** directório com entradas numéricas associadas ao descriptores dos ficheiros abertos.

Info sobre processos (cont.)

- **stat** info mais detalhada sobre a estrutura do processo:

```
1039 (ntpd) S 1 1039 1039 0 -1 320 77 51 358 77 2 1 0 0 8 0 0 100
4453 1953792 475 4294967295 134512640 134713847 3221224480
3221224144 1075070286 0 0 4096 92743 3222544790 0 0 17 0
```

  As entradas desta estrutura são:

  - `%d` o ID do processo
  - `(%s)` o nome do executável do processo
  - `%c` o estado do processo (S = sleeping ou interruptivel ou swapping; R = running; Z = zoombie; ...)
  - `%d` o PID do processo pai
  - `%d` o grupo do processo
  - `%d` o SID do processo
  - ... (vêr mais info nas páginas de manual do `proc`) ...