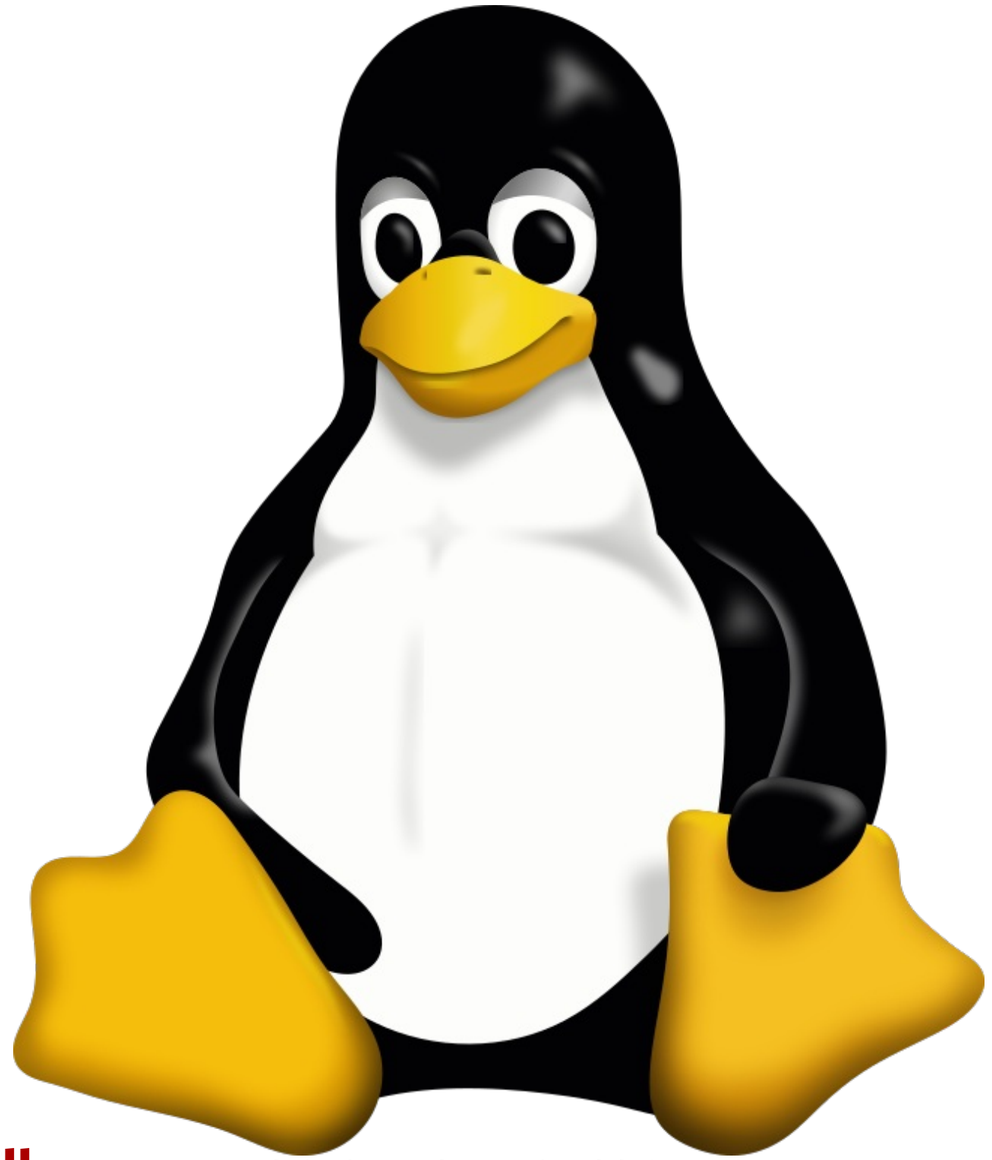
Sistemas Operacionais: explorando o sistema



lewing@isc.tamu.edu Larry Ewing and The GIMP, CC0,
Wikimedia Commons (https://commons.wikimedia.org/wiki/File:Tux.svg)

Prof. Abrantes Araújo Silva Filho

Uso avançado do comando ls

O comando `ls` é um dos mais utilizados pois nos permite ver o que está dentro de um diretório. Ele pode ser utilizado para ver o conteúdo do diretório atual, por exemplo:

```
abrantesasf@server01:/usr/local$ ls
bin  etc  games  include  lib  man  sbin  share  src
```

Mas também pode ser utilizado para ver o que está dentro de outros diretórios que não sejam o diretório de trabalho atual, por exemplo:

```
abrantesasf@server01:/usr/local$ ls /boot/grub
fonts  gfxblacklist.txt  grub.cfg  grubenv  i386-pc  locale  unicode.pf2
```

O diretório no comando acima, `/boot/grub`, é um argumento de linha de comando passado para o comando `ls`. Argumentos de linha de comando são dados/informações que queremos que o comando processe.

Uso avançado do comando ls

Podemos listar até mais de um diretório ao mesmo tempo:

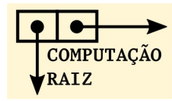
```
abrantesasf@server01:~$ ls /usr/local /boot/grub
/boot/grub:
fonts  gfxblacklist.txt  grub.cfg  grubenv  i386-pc  locale  unicode.pf2

/usr/local:
bin  etc  games  include  lib  man  sbin  share  src
```

Podemos até mudar o formato de saída usando uma opção do comando `ls`, a opção `-l`, antes do argumento:

```
abrantesasf@server01:~$ ls -l /usr/local
total 32
drwxr-xr-x 2 root root 4096 Feb 16 20:51 bin
drwxr-xr-x 2 root root 4096 Feb 16 20:51 etc
drwxr-xr-x 2 root root 4096 Feb 16 20:51 games
drwxr-xr-x 2 root root 4096 Feb 16 20:51 include
drwxr-xr-x 3 root root 4096 Feb 16 20:57 lib
lrwxrwxrwx 1 root root    9 Feb 16 20:51 man -> share/man
drwxr-xr-x 2 root root 4096 Feb 16 20:51 sbin
drwxr-xr-x 7 root root 4096 Aug  4 13:46 share
drwxr-xr-x 2 root root 4096 Feb 16 20:51 src
```

Opções e argumentos dos comandos



A maioria dos comandos aceita "complementos":

- **Opções curtas (uma ou mais, agrupadas ou não)**
- **Opções longas (uma ou mais, não agrupadas)**
- **Argumentos (um ou mais, não agrupados)**

A forma geral de um comando é a seguinte:

```
comando [-opcoes curtas] [--opcoes longas] [argumentos]
```

As opções nos permitem alterar o comportamento do comando para, por exemplo, mostrar mais ou menos informações, mudar o formato de exibição, etc. Como o nome indica, são opcionais.

Os argumentos são os dados/informações sobre os quais o comando vai atuar. Podem ser obrigatórios ou opcionais. É possível misturar opções curtas (agrupadas ou não) com opções longas de diversas formas.

Opções e argumentos dos comandos

Uso de opções curtas não agrupadas:

```
abrantesasf@server01:/usr/local$ ls -l -t
total 32
drwxr-xr-x 7 root root 4096 Aug  4 13:46 share
drwxr-xr-x 3 root root 4096 Feb 16 20:57 lib
drwxr-xr-x 2 root root 4096 Feb 16 20:51 bin
drwxr-xr-x 2 root root 4096 Feb 16 20:51 etc
drwxr-xr-x 2 root root 4096 Feb 16 20:51 games
drwxr-xr-x 2 root root 4096 Feb 16 20:51 include
lrwxrwxrwx 1 root root    9 Feb 16 20:51 man -> share/man
drwxr-xr-x 2 root root 4096 Feb 16 20:51 sbin
drwxr-xr-x 2 root root 4096 Feb 16 20:51 src
```

Opções curtas usadas:

- `-l` mostra o resultado com o formato longo
- `-t` ordena pelo modification time

Opções e argumentos dos comandos

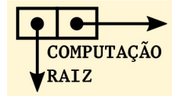
Uso de opções curtas agrupadas:

```
abrantesasf@server01:/usr/local$ ls -lt
total 32
drwxr-xr-x 7 root root 4096 Aug  4 13:46 share
drwxr-xr-x 3 root root 4096 Feb 16 20:57 lib
drwxr-xr-x 2 root root 4096 Feb 16 20:51 bin
drwxr-xr-x 2 root root 4096 Feb 16 20:51 etc
drwxr-xr-x 2 root root 4096 Feb 16 20:51 games
drwxr-xr-x 2 root root 4096 Feb 16 20:51 include
lrwxrwxrwx 1 root root    9 Feb 16 20:51 man -> share/man
drwxr-xr-x 2 root root 4096 Feb 16 20:51 sbin
drwxr-xr-x 2 root root 4096 Feb 16 20:51 src
```

Opções curtas usadas:

- `-l` mostra o resultado com o formato longo
- `-t` ordena pelo modification time

Opções e argumentos dos comandos



Uso de opções longas:

```
abrantesasf@server01:/usr/local$ ls --reverse
src  share  sbin  man  lib  include  games  etc  bin
```

O que a opção acima fez?

Opções e argumentos dos comandos

Uso de opções curtas e longas ao mesmo tempo:

```
abrantesasf@server01:/usr/local$ ls -lt --reverse
total 32
drwxr-xr-x 2 root root 4096 Feb 16 20:51 src
drwxr-xr-x 2 root root 4096 Feb 16 20:51 sbin
lrwxrwxrwx 1 root root    9 Feb 16 20:51 man -> share/man
drwxr-xr-x 2 root root 4096 Feb 16 20:51 include
drwxr-xr-x 2 root root 4096 Feb 16 20:51 games
drwxr-xr-x 2 root root 4096 Feb 16 20:51 etc
drwxr-xr-x 2 root root 4096 Feb 16 20:51 bin
drwxr-xr-x 3 root root 4096 Feb 16 20:57 lib
drwxr-xr-x 7 root root 4096 Aug  4 13:46 share
```

Também é possível usar várias opções curtas misturadas com opções longas.

Obs.: diversas opções podem ser indicadas por uma versão curta ou uma versão longa; outras opções só têm a versão curta ou longa.

Uso avançado do comando ls

O comando `ls` tem 60 opções curtas e/ou longas, para dar a você total flexibilidade na listagem do conteúdo de diretórios.

Isso ilustra muito bem 2 conceitos fundamentais do projeto de sistemas operacionais UNIX e seus derivados:

- **Faça uma coisa só, mas faça bem feito.**

- **Forneça mecanismos, não políticas.**

Uso avançado do comando ls

Algumas opções muito usadas:

```
-a, --all                  do not ignore entries starting with .
-A, --almost-all           do not list implied . and ..
-d, --directory            list directories themselves, not their contents
-F, --classify             append indicator (one of */=>@|) to entries
-h, --human-readable       with -l and -s, print sizes like 1K 234M 2G etc.
-l                         use a long listing format
-r, --reverse              reverse order while sorting
-S                         sort by file size, largest first
-t                         sort by modification time, newest first
```

Uso avançado do comando ls

O formato longo (`-l`) do comando `ls` é muito importante e usado para fornecer mais informações sobre os arquivos e diretórios. É fundamental que você entenda as informações exibidas a seguir:

```
abrantesasf@server01:/etc/ssh$ ls -l
total 652
-rw-r--r-- 1 root root 620042 Jun  9 17:22 moduli
-rw-r--r-- 1 root root   1649 Aug  9  2024 ssh_config
drwxr-xr-x 2 root root   4096 Aug  9  2024 ssh_config.d
-rw-r--r-- 1 root root   3517 Jun  9 17:22 sshd_config
drwxr-xr-x 2 root root   4096 Aug  4 13:49 sshd_config.d
-rw------- 1 root root    505 Aug  4 13:49 ssh_host_ecdsa_key
-rw-r--r-- 1 root root    175 Aug  4 13:49 ssh_host_ecdsa_key.pub
-rw------- 1 root root    399 Aug  4 13:49 ssh_host_ed25519_key
-rw-r--r-- 1 root root     95 Aug  4 13:49 ssh_host_ed25519_key.pub
-rw------- 1 root root   2602 Aug  4 13:49 ssh_host_rsa_key
-rw-r--r-- 1 root root    567 Aug  4 13:49 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root    342 Dec  7  2020 ssh_import_id
```

Uso avançado do comando ls

```
abrantesasf@server01:/etc/ssh$ ls -l
total 652
-rw-r--r-- 1 root root 620042 Jun  9 17:22 moduli
-rw-r--r-- 1 root root   1649 Aug  9  2024 ssh_config
drwxr-xr-x 2 root root   4096 Aug  9  2024 ssh_config.d
-rw-r--r-- 1 root root   3517 Jun  9 17:22 sshd_config
drwxr-xr-x 2 root root   4096 Aug  4 13:49 sshd_config.d
-rw------- 1 root root    505 Aug  4 13:49 ssh_host_ecdsa_key
-rw-r--r-- 1 root root    175 Aug  4 13:49 ssh_host_ecdsa_key.pub
-rw------- 1 root root    399 Aug  4 13:49 ssh_host_ed25519_key
-rw-r--r-- 1 root root     95 Aug  4 13:49 ssh_host_ed25519_key.pub
-rw------- 1 root root   2602 Aug  4 13:49 ssh_host_rsa_key
-rw-r--r-- 1 root root    567 Aug  4 13:49 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root    342 Dec  7  2020 ssh_import_id
```

Tipo de arquivo:

-	**arquivo regular**
d	**diretório**
l	**link simbólico**
b	**dispositivo de bloco**
c	**dispositivo de caractere**
p	**FIFO (named pipe)**
s	**socket**

Uso avançado do comando ls

Caractere	Tipo de arquivo	Explicação
-	Arquivo regular	O tipo mais comum: textos, binários, imagens, etc.
d	Diretório	Contém entradas de arquivos/outros diretórios.
l	Link simbólico	Aponta para outro arquivo (atalho).
b	Dispositivo de bloco	Arquivo especial que fornece acesso bufferizado a dispositivos (ex.: `/dev/sda`).
c	Dispositivo de caractere	Arquivo especial que fornece acesso não bufferizado, caractere a caractere (ex.: `/dev/tty`, `/dev/null`).
p	FIFO (named pipe)	Um pipe nomeado usado para comunicação entre processos.
s	Socket	Usado para comunicação entre processos, local ou pela rede.

Exemplos práticos

- `-rw-r--r--` → arquivo regular de leitura/escrita.
- `drwxr-xr-x` → diretório.
- `lrwxrwxrwx` → link simbólico.
- `crw-rw-rw-` → dispositivo de caractere (ex.: terminal).
- `brw-rw----` → dispositivo de bloco (ex.: HD, SSD).
- `prw-------` → FIFO.
- `srwxr-xr-x` → socket UNIX.

Uso avançado do comando ls

```
abrantesasf@server01:/etc/ssh$ ls -l
total 652
-rw-r--r-- 1 root root 620042 Jun  9 17:22 moduli
-rw-r--r-- 1 root root   1649 Aug  9  2024 ssh_config
drwxr-xr-x 2 root root   4096 Aug  9  2024 ssh_config.d
-rw-r--r-- 1 root root   3517 Jun  9 17:22 sshd_config
drwxr-xr-x 2 root root   4096 Aug  4 13:49 sshd_config.d
-rw------- 1 root root    505 Aug  4 13:49 ssh_host_ecdsa_key
-rw-r--r-- 1 root root    175 Aug  4 13:49 ssh_host_ecdsa_key.pub
-rw------- 1 root root    399 Aug  4 13:49 ssh_host_ed25519_key
-rw-r--r-- 1 root root     95 Aug  4 13:49 ssh_host_ed25519_key.pub
-rw------- 1 root root   2602 Aug  4 13:49 ssh_host_rsa_key
-rw-r--r-- 1 root root    567 Aug  4 13:49 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root    342 Dec  7  2020 ssh_import_id
```

Permissões de acesso ao arquivo: 3 grupos de 3 letras mostrando as permissões:

- **1º grupo de 3 letras: permissões do dono do arquivo**
- **2º grupo de 3 letras: permissões do usuários do grupo do arquivo**
- **3º grupo de 3 letras: permissões de outros usuários**

As letras podem ser: `r` (leitura), `w` (gravação), `x` (execução), `s/S` (setuid e/ou setgid com/sem `x`), `t/T` (stick com/sem `x`), e `-`

Uso avançado do comando ls

```
abrantesasf@server01:/etc/ssh$ ls -l
total 652
-rw-r--r-- 1 root root 620042 Jun  9 17:22 moduli
-rw-r--r-- 1 root root   1649 Aug  9  2024 ssh_config
drwxr-xr-x 2 root root   4096 Aug  9  2024 ssh_config.d
-rw-r--r-- 1 root root   3517 Jun  9 17:22 sshd_config
drwxr-xr-x 2 root root   4096 Aug  4 13:49 sshd_config.d
-rw------- 1 root root    505 Aug  4 13:49 ssh_host_ecdsa_key
-rw-r--r-- 1 root root    175 Aug  4 13:49 ssh_host_ecdsa_key.pub
-rw------- 1 root root    399 Aug  4 13:49 ssh_host_ed25519_key
-rw-r--r-- 1 root root     95 Aug  4 13:49 ssh_host_ed25519_key.pub
-rw------- 1 root root   2602 Aug  4 13:49 ssh_host_rsa_key
-rw-r--r-- 1 root root    567 Aug  4 13:49 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root    342 Dec  7  2020 ssh_import_id
```

Número de hard links ao arquivo

Uso avançado do comando ls

```
abrantesasf@server01:/etc/ssh$ ls -l
total 652
-rw-r--r-- 1 root root 620042 Jun  9 17:22 moduli
-rw-r--r-- 1 root root   1649 Aug  9  2024 ssh_config
drwxr-xr-x 2 root root   4096 Aug  9  2024 ssh_config.d
-rw-r--r-- 1 root root   3517 Jun  9 17:22 sshd_config
drwxr-xr-x 2 root root   4096 Aug  4 13:49 sshd_config.d
-rw------- 1 root root    505 Aug  4 13:49 ssh_host_ecdsa_key
-rw-r--r-- 1 root root    175 Aug  4 13:49 ssh_host_ecdsa_key.pub
-rw------- 1 root root    399 Aug  4 13:49 ssh_host_ed25519_key
-rw-r--r-- 1 root root     95 Aug  4 13:49 ssh_host_ed25519_key.pub
-rw------- 1 root root   2602 Aug  4 13:49 ssh_host_rsa_key
-rw-r--r-- 1 root root    567 Aug  4 13:49 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root    342 Dec  7  2020 ssh_import_id
```

Usuário que é o dono do arquivo

Uso avançado do comando ls

```
abrantesasf@server01:/etc/ssh$ ls -l
total 652
-rw-r--r-- 1 root root 620042 Jun  9 17:22 moduli
-rw-r--r-- 1 root root   1649 Aug  9  2024 ssh_config
drwxr-xr-x 2 root root   4096 Aug  9  2024 ssh_config.d
-rw-r--r-- 1 root root   3517 Jun  9 17:22 sshd_config
drwxr-xr-x 2 root root   4096 Aug  4 13:49 sshd_config.d
-rw------- 1 root root    505 Aug  4 13:49 ssh_host_ecdsa_key
-rw-r--r-- 1 root root    175 Aug  4 13:49 ssh_host_ecdsa_key.pub
-rw------- 1 root root    399 Aug  4 13:49 ssh_host_ed25519_key
-rw-r--r-- 1 root root     95 Aug  4 13:49 ssh_host_ed25519_key.pub
-rw------- 1 root root   2602 Aug  4 13:49 ssh_host_rsa_key
-rw-r--r-- 1 root root    567 Aug  4 13:49 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root    342 Dec  7  2020 ssh_import_id
```

Grupo proprietário do arquivo

Uso avançado do comando ls

```
abrantesasf@server01:/etc/ssh$ ls -l
total 652
-rw-r--r-- 1 root root 620042 Jun  9 17:22 moduli
-rw-r--r-- 1 root root   1649 Aug  9  2024 ssh_config
drwxr-xr-x 2 root root   4096 Aug  9  2024 ssh_config.d
-rw-r--r-- 1 root root   3517 Jun  9 17:22 sshd_config
drwxr-xr-x 2 root root   4096 Aug  4 13:49 sshd_config.d
-rw------- 1 root root    505 Aug  4 13:49 ssh_host_ecdsa_key
-rw-r--r-- 1 root root    175 Aug  4 13:49 ssh_host_ecdsa_key.pub
-rw------- 1 root root    399 Aug  4 13:49 ssh_host_ed25519_key
-rw-r--r-- 1 root root     95 Aug  4 13:49 ssh_host_ed25519_key.pub
-rw------- 1 root root   2602 Aug  4 13:49 ssh_host_rsa_key
-rw-r--r-- 1 root root    567 Aug  4 13:49 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root    342 Dec  7  2020 ssh_import_id
```

Tamanho do arquivo, em bytes.

Obs.: aquela linha onde está escrito "`total 652`" não é a soma total do tamanho em bytes dos arquivos. Em geral indica o tamanho dos blocos de disco, em unidades de 1 KB, usados pelos arquivos listados no diretório.

Sim, isso é confuso até mesmo para usuários mais experientes, ainda mais porque o tamanho real de cada bloco físico em um disco costuma ser de 4 KB.

Uso avançado do comando ls

```
abrantesasf@server01:/etc/ssh$ ls -l
total 652
-rw-r--r-- 1 root root 620042 Jun  9 17:22 moduli
-rw-r--r-- 1 root root   1649 Aug  9  2024 ssh_config
drwxr-xr-x 2 root root   4096 Aug  9  2024 ssh_config.d
-rw-r--r-- 1 root root   3517 Jun  9 17:22 sshd_config
drwxr-xr-x 2 root root   4096 Aug  4 13:49 sshd_config.d
-rw------- 1 root root    505 Aug  4 13:49 ssh_host_ecdsa_key
-rw-r--r-- 1 root root    175 Aug  4 13:49 ssh_host_ecdsa_key.pub
-rw------- 1 root root    399 Aug  4 13:49 ssh_host_ed25519_key
-rw-r--r-- 1 root root     95 Aug  4 13:49 ssh_host_ed25519_key.pub
-rw------- 1 root root   2602 Aug  4 13:49 ssh_host_rsa_key
-rw-r--r-- 1 root root    567 Aug  4 13:49 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root    342 Dec  7  2020 ssh_import_id
```

Data e hora da última modificação do arquivo. O formato é variável, mas pode ser padronizado com um opção.

Uso avançado do comando ls

```
abrantesasf@server01:/etc/ssh$ ls -l
total 652
-rw-r--r-- 1 root root 620042 Jun  9 17:22 moduli
-rw-r--r-- 1 root root   1649 Aug  9  2024 ssh_config
drwxr-xr-x 2 root root   4096 Aug  9  2024 ssh_config.d
-rw-r--r-- 1 root root   3517 Jun  9 17:22 sshd_config
drwxr-xr-x 2 root root   4096 Aug  4 13:49 sshd_config.d
-rw------- 1 root root    505 Aug  4 13:49 ssh_host_ecdsa_key
-rw-r--r-- 1 root root    175 Aug  4 13:49 ssh_host_ecdsa_key.pub
-rw------- 1 root root    399 Aug  4 13:49 ssh_host_ed25519_key
-rw-r--r-- 1 root root     95 Aug  4 13:49 ssh_host_ed25519_key.pub
-rw------- 1 root root   2602 Aug  4 13:49 ssh_host_rsa_key
-rw-r--r-- 1 root root    567 Aug  4 13:49 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root    342 Dec  7  2020 ssh_import_id
```

Nome do arquivo.

Qual o tipo do arquivo?

Considere os arquivos abaixo. Diga qual deles é uma imagem e qual deles é um arquivo de texto:

```
abrantesasf@server01:~$ ls
apport.png   apport.txt
```

Qual o tipo do arquivo?

O UNIX/Linux não utiliza extensões para determinar o tipo de arquivos, o mecanismo é muito mais sofisticado. Podemos usar o comando `file` para determinar o tipo de um arquivo.

```
abrantesasf@server01:/$ ls
bin                  home                 mnt   sbin.usr-is-merged  usr
bin.usr-is-merged    lib                  opt   snap                var
boot                 lib64                proc  srv
cdrom                lib.usr-is-merged    root  swap.img
dev                  lost+found           run   sys
etc                  media                sbin  tmp
abrantesasf@server01:/$ file swap.img
swap.img: regular file, no read permission
abrantesasf@server01:/$ file tmp
tmp: sticky, directory
abrantesasf@server01:/$ file home
home: directory
abrantesasf@server01:/$ file lib
lib: symbolic link to usr/lib
abrantesasf@server01:/$ file boot/grub/grub.cfg
boot/grub/grub.cfg: regular file, no read permission
abrantesasf@server01:/$ _
```

Qual o tipo do arquivo?

O arquivo que parece uma imagem PNG é um arquivo texto; o arquivo que parece um arquivo TXT é uma imagem. As extensões não importam para o tipo de arquivo! Não são nem necessárias!

```
abrantesasf@server01:~$ ls
apport.png  apport.txt
abrantesasf@server01:~$ file apport.png
apport.png: Unicode text, UTF-8 text
abrantesasf@server01:~$ file apport.txt
apport.txt: PNG image data, 64 x 64, 8-bit/color RGBA, non-interlaced
```

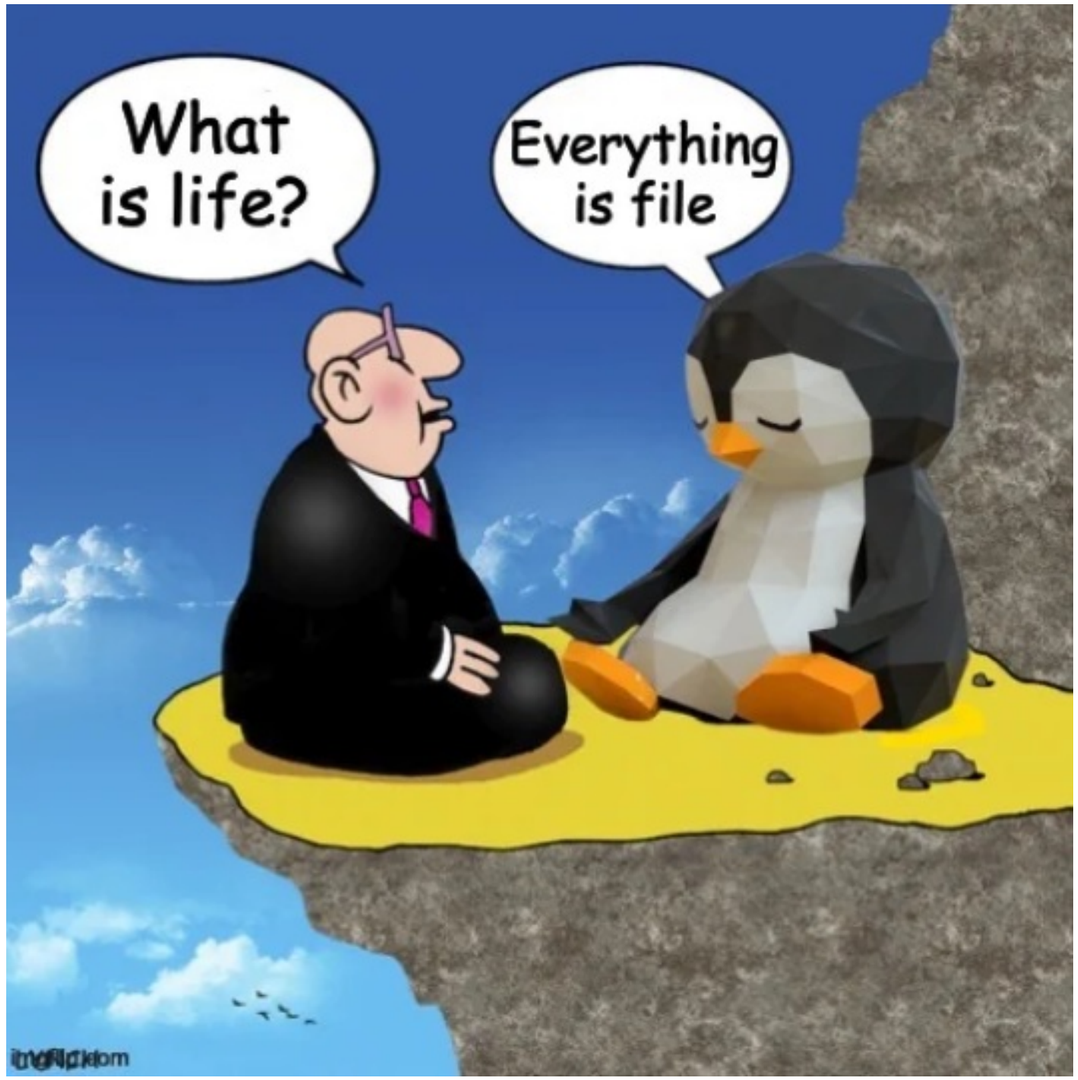
Tudo é arquivo

Um conceito fundamental nos sistemas UNIX, Linux e derivados é que TUDO É ARQUIVO. Isso significa que o sistema operacional trata praticamente todos os recursos e objetos como um arquivo, incluindo diretórios, processos, dispositivos e até mesmo o próprio sistema de arquivos.

Essa abordagem simplifica muito a interação com o sistema, pois permite usar as mesmas ferramentas e comandos para manipular diferentes tipos de arquivos. Por exemplo, você pode usar o comando `ls` para listar arquivos, diretórios e até mesmo processos em execução.

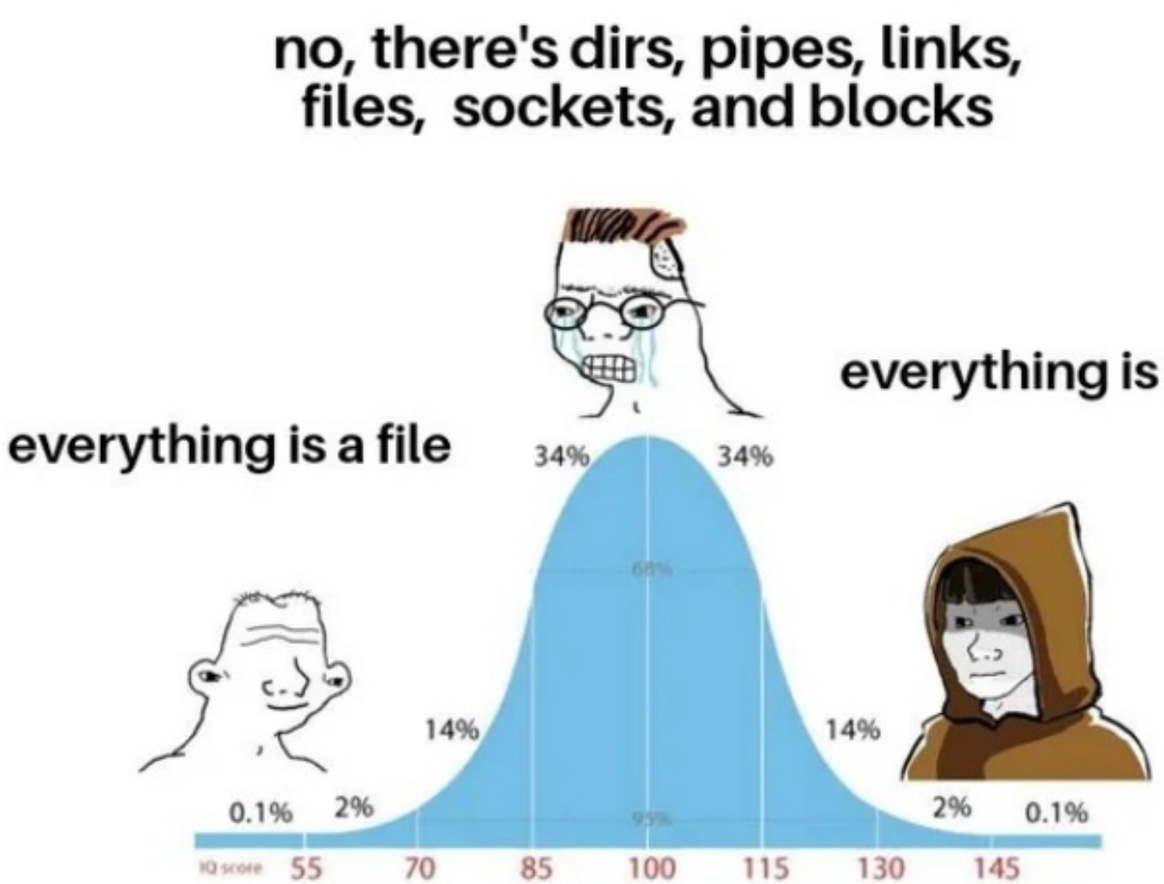
Tudo é arquivo

Tudo é arquivo



Tudo é arquivo

Tudo é arquivo



Como visualizar o conteúdo de um arquivo? Use o less!

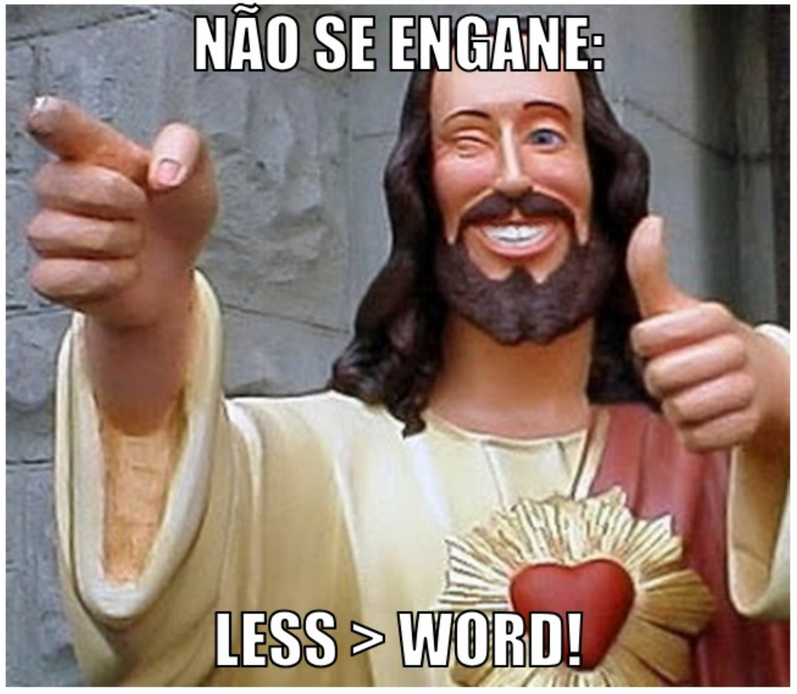
Diversos comandos podem ser utilizados para ver o conteúdo de um arquivo de texto. Dentre eles, os comandos `more` e `less` são os mais utilizados (principalmente o `less`).

```
abrantesasf@server01:~$ less /etc/passwd
```

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin
```

Uso básico do less

`b, PAGE UP`	voltar 1 página
`space, PAGE DOWN`	avançar 1 página
`UP ARROW`	voltar 1 linha
`DOWN ARROW`	avançar 1 linha
`G`	vai para o final do arquivo
`1G, g`	vai para o início do arquivo
`/string`	pesquisa por string
`n`	busca próxima ocorrência de string
`q`	sai



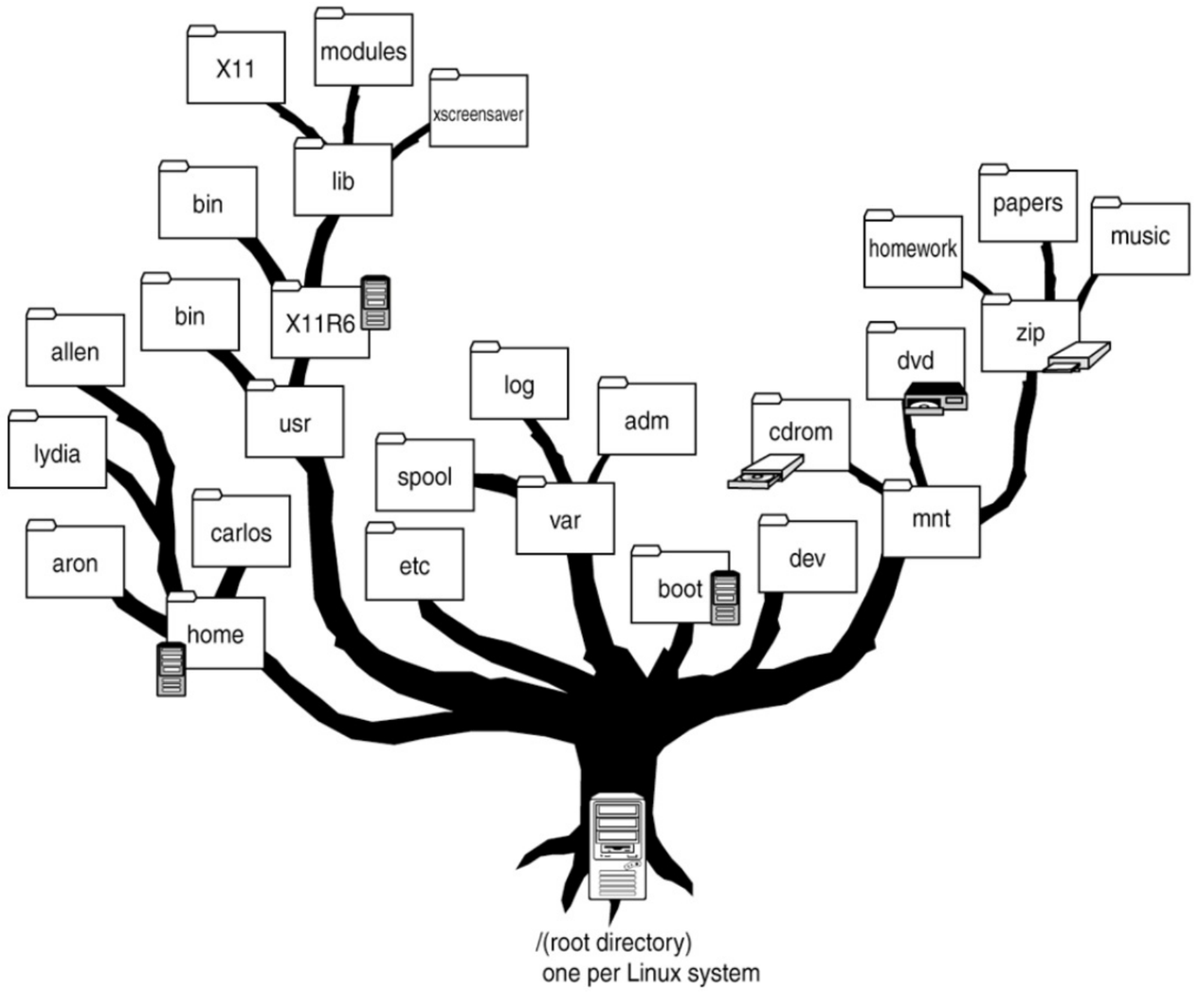
Obs.: programas como o `less` e o `more` pertencem a uma classe de programas chamados de **pagers**, que permitem a visualização de longos documentos de texto, de modo página a página.

O que é um arquivo em texto puro?

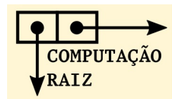
Arquivos em texto puro contém apenas caracteres (ASCII, UTF-8, etc).
Arquivos do Word ou LibreOffice contém elementos não textuais para estrutura, formatação, cores, etc.

O "Linux Filesystem Hierarhcy Standard"

É um padrão que diz como os arquivos e diretórios em um sistema Linux devem ser organizados. Não é 100% seguido por todas as distribuições, mas é quase!



O "Linux Filesystem Hierarhcy Standard"



Use comandos como o `cd`, `ls`, `file` e `less` para visitar todos os diretórios abaixo. Familiarize-se com os diretórios e conteúdos! Não tenha medo (use o `reset` se precisar restaurar o estado do console/terminal virtual).

`/` diretório raiz, pai de todos os outros

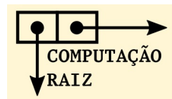
`/bin` binários que devem estar presentes para o sistema iniciar e rodar

`/boot` contém o kernel do Linux, imagem RAM inicial (para drivers durante o boot) e o boot loader. Arquivos interessantes:

`/boot/vmlinuz` `/boot/grub/grub.cfg`
`/boot/initrd.img`

`/dev` contém arquivos que representam dispositivos (devines nodes); é a lista e configuração dos dispositivos que o kernel entende

O "Linux Filesystem Hierarhcy Standard"



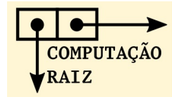
`/etc` um dos diretórios mais importantes, pois contém todos os arquivos de configurações globais do sistema, os shell scripts que inicializam os processos e serviços durante o boot, e muito mais. Tudo neste diretório deve ser arquivo texto puro. Tudo aqui é interessante, mas alguns favoritos são:

```
/etc/crontab
/etc/fstab
/etc/passwd
```

`/home` armazena os diretórios HOME dos usuários, um dos únicos diretórios que os usuários têm permissão para fazer o que quiserem. Cada usuário tem o seu próprio diretório HOME.

`/lib` bibliotecas compartilhadas usadas pelos programas mais fundamentais do sistema

O "Linux Filesystem Hierarhcy Standard"



`/lost+found`

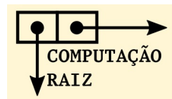
toda partição formatada ou dispositivo usando um sistema de arquivos Linux terá um dirertório com esse nome, e é usado em situações de recuperação em alguma situação de corrupção do sistema de arquivos. Se nada de ruim ocorrer no sistema, fica vazio.

`/media` **ponto de montagem para dispositívos removíveis (USB, CD-ROM, etc.) que são montados automaticamente ao serem inseridos.**

`/mnt` **ponto de montagem de dispositivos (removíveis ou não) que serão montados manualmente**

`/opt` **software "opcional", de terceiros, comerciais ou não, instalados pelo próprio usuário**

O "Linux Filesystem Hierarhcy Standard"



`/proc` sistema virtual mantido pelo kernel; os arquivos nesse sistema fornecem informações sobre o kernel, processos, hardware, etc. Alguns arquivos muito interessantes são:

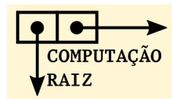
`/proc/cpuinfo` `/proc/meminfo`
`/proc/version` `/proc/swaps`

`/root` diretório HOME do usuário root

`/sbin` binários de programas de sistemas, que geralmente são reservados para o usuário root

`/tmp` armazenamento temporário de arquivos criados por diversos programas ou usuários, podendo ser configurado para ser apagado de maneira automática durante o boot do sistema (não use para nada que precise ser armazenado de forma permanente!)

O "Linux Filesystem Hierarhcy Standard"



`/usr` provavelmente é o maior de todos os diretórios em um sistema Linux, e contém todos os programas e arquivos de configuração e suporte que são usados pelos usuários regulares do sistema

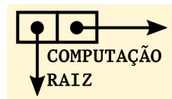
`/usr/bin` programas instalados pela distribuição Linux para seus usuários

`/usr/lib` bibliotecas compartilhadas para os programas em **`/usr/bin`**

`/usr/local`
armazena programas que não são instalados pela distribuição Linux, mas que serão utilizados de modo global no sistema. Programas assim costumam ser instalados em **`/usr/local/bin`** e outros subdiretórios.

`/usr/sbin` mais programas administrativos para o usuário root

O "Linux Filesystem Hierarhcy Standard"



`/usr/share`

todos os dados compartilhados utilizados pelos programas em `/usr/bin`. Contém coisas como arquivos de configuração, ícones, background de telas gráficas, arquivos de sons, etc.

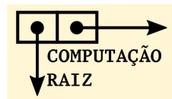
`/usr/share/doc`

a maioria dos pacotes instalados no sistema automaticamente instala sua documentação para consulta nesse subdiretório, em diversos formatos (txt, pdf, html, man, etc.)

`/var` **armazena dados que são alterados frequentemente, como bancos de dados, logs, arquivos de spool, arquivos de email, etc.**

`/var/log` **armazena os logs do sistema e de diversos outros serviços, essencial para a administração e debug do sistema operacional e de serviços.**

O "Linux Filesystem Hierarhcy Standard"



Existem muitos outros diretórios em um sistema UNIX ou Linux. A melhor maneira de você aprender é visitar os diretórios, listar o conteúdo e dar uma olhada em alguns dos arquivos.

Não se preocupe em estragar o sistema, se você está usando um usuário normal não privilegiado (usuário diferente de root, sem permissões adicionais), você não conseguirá estragar o sistema (bom, talvez só o seu próprio diretório HOME).

Se você acidentalmente abrir um arquivo binário e o seu terminal ficar "bugado", não se preocupe: apenas use o comando `reset` para seu terminal ser reiniciado e voltar ao normal.

Links simbólicos

Um tipo extremamente útil de arquivo é o chamado link simbólico (symbolic link, soft link, symlink). Ele é, na verdade, uma espécie de "apelido" para um arquivo ou diretório. Isso nos dá uma capacidade enorme no sistema. Podemos criar:

- Diferentes nomes para o mesmo arquivo/diretório

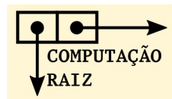
- Um nome que aponta para versões específicas de um arquivo

```
abrantesasf@server01:/$ ls -l bin lib lib64 sbin
lrwxrwxrwx 1 root root 7 Apr 22  2024 bin -> usr/bin
lrwxrwxrwx 1 root root 7 Apr 22  2024 lib -> usr/lib
lrwxrwxrwx 1 root root 9 Apr 22  2024 lib64 -> usr/lib64
lrwxrwxrwx 1 root root 8 Apr 22  2024 sbin -> usr/sbin
```

Excelente para criar atalhos para diretórios e arquivos!

Existem no UNIX desde 1977. O Windows só consegui reproduzir essa funcionalidade em 2007 (Windows Vista).

Hard links



É outra maneira de criar múltiplos nomes para um mesmo arquivo, mas isso é feito de forma diferente do link simbólico.

Não é muito mais usado hoje em dia mas, por razões históricas, o comando "`ls -l`" mostra o número de hard links para um arquivo.