

Manipulação de arquivos de texto



Prof. Dr. João Paulo Lemos Escola
Copyright © 2022

Conteúdo

- touch
- stat
- head
- tail
- grep
- find
- awk
- for
- sort
- uniq
- diff

Gerando um arquivo

- O comando *touch* cria um arquivo vazio:

```
ubuntu@ubuntu:~$ touch a.txt
ubuntu@ubuntu:~$ ls
a.txt      Documentos  Imagens    Música      Público    Vídeos
Desktop    Downloads  Modelos    programa.sh snap
ubuntu@ubuntu:~$ cat a.txt
ubuntu@ubuntu:~$
```

- Com o comando *ls* vemos o arquivo que foi criado e com o comando *cat* podemos ver seu conteúdo (vazio).

stat

- Vamos adicionar conteúdo no arquivo com o comando *ps aux > a.txt*;
- Agora vamos ver o status do arquivo:

```
ubuntu@ubuntu:~$ ps aux > a.txt
ubuntu@ubuntu:~$ stat a.txt
Arquivo: a.txt
  Tamanho: 17942      Blocos: 40      bloco de E/S: 4096  arquivo comum
Dispositivo: 1ch/28d  Inode: 3781     Links: 1
Acesso: (0664/-rw-rw-r--)  Uid: ( 999/  ubuntu)  Gid: ( 999/  ubuntu)
Acesso: 2022-07-21 14:51:05.457385117 +0000
Modificação: 2022-07-21 14:51:05.473435329 +0000
  Alteração: 2022-07-21 14:51:05.473435329 +0000
  Criação: -
ubuntu@ubuntu:~$
```

head

- Mostra as 10 primeiras linhas do arquivo:

```
ubuntu@ubuntu:~$ head a.txt
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME
root           1   0.0   0.3 168080   6464 ?        Ss   09:25   0:02
ybe-ubiquity splash ---
root           2   0.0   0.0     0     0 ?        S    09:25   0:00
root           3   0.0   0.0     0     0 ?        I<   09:25   0:00
root           4   0.0   0.0     0     0 ?        I<   09:25   0:00
root           6   0.0   0.0     0     0 ?        I<   09:25   0:00
-events_highpri]
root           9   0.0   0.0     0     0 ?        I<   09:25   0:00
]
root          10   0.0   0.0     0     0 ?        S    09:25   0:00
de_]
root          11   0.0   0.0     0     0 ?        S    09:25   0:00
ace]
root          12   0.0   0.0     0     0 ?        S    09:25   0:01
ubuntu@ubuntu:~$
```

Parâmetros do head

- Com o parâmetro `-n` podemos especificar o número de linhas que serão retornadas:

```
ubuntu@ubuntu:~$ head -n 1 a.txt
USER          PID %CPU %MEM    VSZ   RSS TTY
ubuntu@ubuntu:~$
```

```
ubuntu@ubuntu:~$ head -n 2 a.txt
USER          PID %CPU %MEM    VSZ   RSS TTY
root           1  0.0  0.3 168080 6464 ?
```

```
ubuntu@ubuntu:~$ head -n 3 a.txt
USER          PID %CPU %MEM    VSZ   RSS TTY
root           1  0.0  0.3 168080 6464 ?
root           2  0.0  0.0      0     0 ?
```

tail

- Mostra as 10 últimas linhas de um arquivo, ou as N últimas se utilizarmos o parâmetro *-n*:

```
ubuntu@ubuntu:~$ tail -n 1 a.txt
ubuntu      11291  0.0  0.0  21408  1552 pts/0
ubuntu@ubuntu:~$ tail -n 2 a.txt
root        11249  0.0  0.0    0      0 ?
ubuntu      11291  0.0  0.0  21408  1552 pts/0
ubuntu@ubuntu:~$ tail -n 3 a.txt
root        11178  0.0  0.0    0      0 ?
root        11249  0.0  0.0    0      0 ?
ubuntu      11291  0.0  0.0  21408  1552 pts/0
```

Uso do caractere pipe

- O caractere pipe serve para filtrar o conteúdo, concatenando comandos;
- Aqui estamos mostrando o conteúdo do arquivo com o comando *cat* e, em seguida, especificando que queremos o conteúdo os N últimos caracteres:

```
ubuntu@ubuntu:~$ cat a.txt | tail -1
ubuntu      11291  0.0  0.0  21408  1552 pts/0
ubuntu@ubuntu:~$ cat a.txt | tail -2
root        11249  0.0  0.0      0      0 ?
ubuntu      11291  0.0  0.0  21408  1552 pts/0
ubuntu@ubuntu:~$ cat a.txt | tail -3
root        11178  0.0  0.0      0      0 ?
root        11249  0.0  0.0      0      0 ?
ubuntu      11291  0.0  0.0  21408  1552 pts/0
```


Pipe com head

- Mesma aplicação do slide anterior, entretanto agora usando o *head*:

```
ubuntu@ubuntu:~$ cat a.txt | head -1
USER          PID %CPU %MEM    VSZ   RSS TTY
ubuntu@ubuntu:~$ cat a.txt | head -2
USER          PID %CPU %MEM    VSZ   RSS TTY
root           1  0.0  0.3 168080  6464 ?
ubuntu@ubuntu:~$ cat a.txt | head -5
USER          PID %CPU %MEM    VSZ   RSS TTY
root           1  0.0  0.3 168080  6464 ?
root           2  0.0  0.0      0      0 ?
root           3  0.0  0.0      0      0 ?
root           4  0.0  0.0      0      0 ?
```

Procurando texto no arquivo

- O comando *grep* permite procurar um termo no conteúdo do arquivo:

```
ubuntu@ubuntu:~$ grep calendar a.txt
ubuntu      7179  0.0  0.1 582864  2856 ?        Sl    09:48   0:00 /usr/libexec/gnome-shell-calendar-ser
ver
ubuntu      7195  0.0  0.2 719036  5116 ?        Ssl   09:48   0:00 /usr/libexec/evolution-calendar-facto
ry
ubuntu      7583  0.0  0.8 735876 16612 ?        Sl    09:48   0:00 /usr/bin/gnome-calendar --gapplicatio
```

grep

- Também nos permite procurar arquivos que contenham determinado texto ou caracteres:

```
ubuntu@ubuntu:~$ echo banana > b.txt
ubuntu@ubuntu:~$ echo abacate > c.txt
ubuntu@ubuntu:~$ grep -lir banana .
./b.txt
./snap/snap-store/common/.cache/gnome-software
./cache/tracker3/files/http%3A%2F%2Ftracker.
ubuntu@ubuntu:~$ grep -lir aba .
./c.txt
```

find

- Permite procurar arquivos pelo nome ou parte do nome:

```
ubuntu@ubuntu:~$ find . -iname "*.txt"
./c.txt
./b.txt
./a.txt
./.cache/tracker3/files/last-crawl.txt
./.cache/tracker3/files/locale-for-miner-apps.txt
./.cache/tracker3/files/first-index.txt
```

awk

- Poderoso comando que permite, neste caso, filtrar por coluna:

```
ubuntu@ubuntu:~$ awk '{print $1}' a.txt
USER
root
root
root
root
root
root
root
root
```

```
ubuntu@ubuntu:~$ awk '{print $2}' a.txt | head
PID
1
2
3
4
6
9
10
11
12
ubuntu@ubuntu:~$ awk '{print $3}' a.txt | head
%CPU
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
0.0
```

WC

- Word count: retorna o número de linhas, palavras e caracteres de um ou mais arquivos:

```
ubuntu@ubuntu:~$ wc a.txt
195  2257 18714 a.txt
ubuntu@ubuntu:~$ wc -l a.txt
195 a.txt
ubuntu@ubuntu:~$ wc -l *.txt
195 a.txt
   1 b.txt
   1 c.txt
197 total
```

Iteração FOR

- Podemos criar laços For ou While no terminal do Linux;
- Dentro do laço, podemos manipular cada arquivo conforme nossa necessidade:

```
ubuntu@ubuntu:~$ for i in *.txt; do ls $i; done
a.txt
b.txt
c.txt
ubuntu@ubuntu:~$ for i in *.txt; do echo 1 >> $i; done
ubuntu@ubuntu:~$ wc *.txt
 196  2258 18716 a.txt
   2     2     9 b.txt
   2     2    10 c.txt
 200  2262 18735 total
```

sort

- Mostra o conteúdo ordenado;
- O parâmetro *-r* inverte a ordenação:

```
ubuntu@ubuntu:~$ sort -r lista.txt
banana
amora
abacaxi
abacaxi
abacaxi
abacaxi
abacaxi
ubuntu@ubuntu:~$
```


uniq

- Mostra os dados, eliminando repetições:

```
ubuntu@ubuntu:~$ uniq lista.txt
abacaxi
amora
banana
ubuntu@ubuntu:~$
```

diff

- Mostra as diferenças entre os conteúdos dos arquivos:

```
ubuntu@ubuntu:~$ cp lista.txt lista2.txt
ubuntu@ubuntu:~$ diff lista.txt lista2.txt
ubuntu@ubuntu:~$
```

```
ubuntu@ubuntu:~$ diff lista.txt lista2.txt
7a8
> laranja
ubuntu@ubuntu:~$ diff lista2.txt lista.txt
8d7
< laranja
ubuntu@ubuntu:~$
```

O que vimos?

- touch
- stat
- head
- tail
- grep
- find
- awk
- for
- sort
- uniq
- diff

Na próxima aula...

- Comandos de rede.