COMPACTANDO E DESCOMPACTANDO

Linux

Função zip:

Para zipar um diretório no linux, devemos utilizar os comandos:

zip (nome_que_queremos.zip) diretorio.zip (nome real do diretorio) diretorio

Após isso, será feita a compactação do diretório. Para adicionarmos os arquivos dentro diretorio, devemos usar recursão -r (zip -r diretorio.zip diretorio)

Teste-o: Apague o diretorio original e após isso, vamos descompactar. Para ver os arquivos dentro, use o comando unzip -l (nome_arquivo_zip). Para apenas descompactar use o unzip e ele fará a descompactação completa do arquivo zip. Caso deseje deixar a tela mais "limpa", use o comando: unzip -q (quiet)

Para, por exemplo, realizarmos o zip (dentro de um diretório) de todos os arquivos txt, faremos: zip (nome do zip.zip) *(significa todos) .txt (ou seja, todos os arquivos txt desse diretório)

zip bemvindo.zip *.txt
Para criar um arquivo no formato .zip utilizamos o commando zip passando o nome do arquivo que desejamos criar e os arquivos que desejamos incluir. No nosso caso todos que possuem extensão .txt.

Para listar os arquivos dentro do arquivo .zip utilizamos o comando unzip passando o parâmetro -l:

$ unzip -l work.zip

Função TAR:

tar -cz workspace > work.tar.gz

O tar sozinho não serve para compactar arquivos. Na verdade o tar serve para empacotar vários diretórios e arquivos em um único arquivo, facilitando a transferência. Para compactar usaremos o tar combinado com o zip, o primeiro empacota e o segundo compacta.

O modificador -cz indica que o arquivo *tar* será criado (-c) e será compactado pelo *zip*(-z) usando o redirecionamento >. Uma observação interessante é que comando tar já é automaticamente recursivo.

Lembre-se de utilizar o comando ls para verificar o arquivo criado e de remover o diretório workspace antes de descompactar os arquivos com o comando rm.

tar -c(create)z(zip) workspace(diretorio) (nome diretorio) >(direcionamento para o arquivo tar que queremos criar) arquivo.tar.gz

Para descompactar o arquivo .tar.gz que criamos, fazemos:

tar -xz <workspace.tar.gz

x é para extrair o arquivo, z para zipar e < para entrada de dados

>	Redireciona saída
<	Redireciona entrada

Podemos realizar também, de forma mais fácil, o uso do -f para por nome no arquivo que queremos…

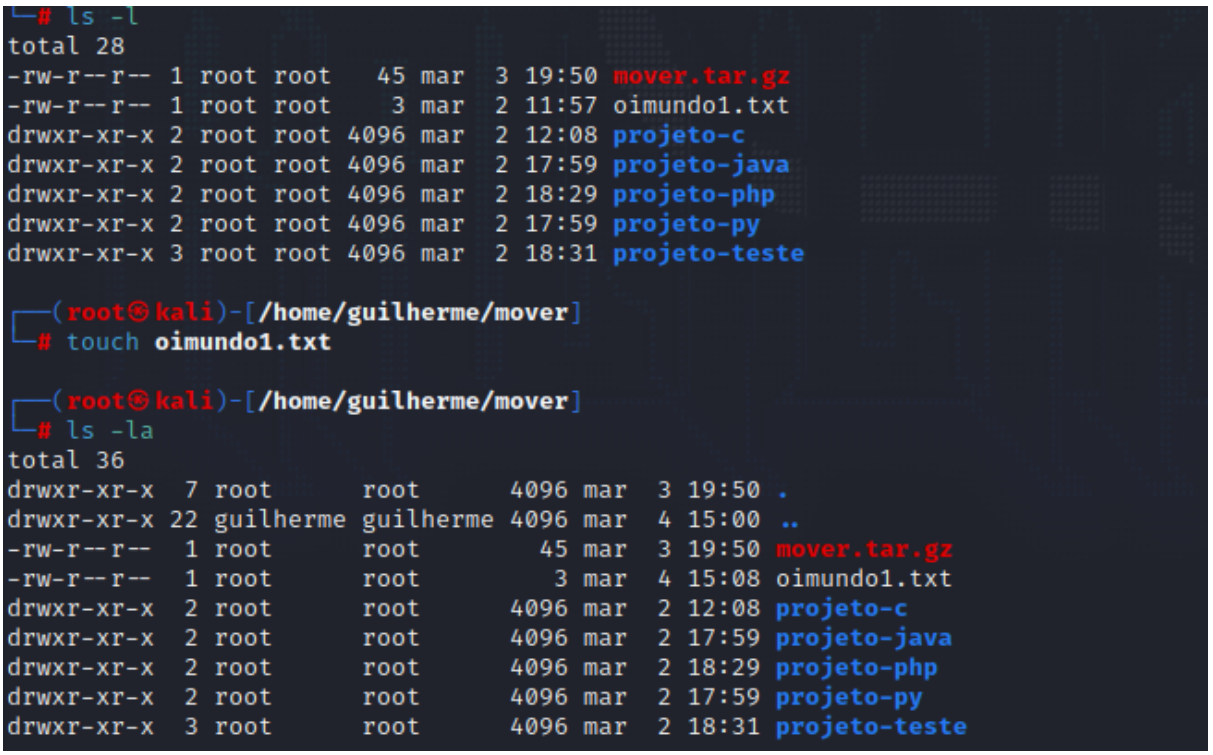
tar -czf work.tar.gz workspace/

E o comando vai ficar primeiro o nome do arquivo que queremos criar e depois o diretório.

Outra excelente dica é usar o -v(FLAG), de verbose (para mostrar o que tem dentro daquele tar). Seu uso fica: tar -v(c ou x)zf xx.tar.gz x.

Touch, Data da última Modificação e Data do sistema:

O comando touch permite a criação de arquivos txt e a modificação de data e hora de arquivo. Por exemplo, veja abaixo:



Fizemos a mudança da data e horário do oimundo1.txt para hoje, 04/03 15:08. Não modificamos o conteúdo do arquivo, mas sim a data de modificação. Use date para testar a data do seu Linux.

Less, head, tail:

O comando head permite as 10 primeiras linhas de um arquivo. Caso queiramos mais ou menos números de linhas, poderemos utilizar o -n (número de linhas)

head -n 3 google.txt (imprima as 3 linhas do arquivo)

O comando tail serve para ler o final do arquivo (rodapé). Pode usar o -n nele também.

Outra dica de leitura é o uso do less para leitura de algo. Para movimentação, use as setas e para sair do programa, use o “q”.

Edição de arquivos com o vi:

Comandos do vi: Navegação com as setas do teclado. Para escrever algo, aperte a letra i(insert). Para sair do modo de edição, aperte ESC e você será retornado ao modo de navegação. Para salvar aperte shift : (dois pontos) e digite w para salvar. Para salvar e sair, digite wq. w: Whrite / / q: quit

Outras letras utilizadas no vi:

“a”: Próxima palavra ao lado.

“x”: Para remoção de um caractere. Dica: para apagar a quantidade de caracteres que você deseja, digite o número antes e depois aperte x (sempre deixe o curso antes do que você quer apagar).

“dd”: Para remoção de uma linha completa! Ponha no início da linha.

“SHIFT+a”: irá para o final da linha.

“SHIFT+g” ou só “G”: irá para a última linha. Funciona também para navegação (linha 30… 30 shift g)

“g”: Início do texto. Pode por números antes para ver onde deseja ir, ex: 1g,4g,gg

“$”: Para o final da linha atual.

“0”: Primeiro caractere da linha, início da linha.

Se digitarmos /, serve para pesquisar alguma palavra. Ex: /software(ele me joga para essa linha da palavra) e para olharmos as outras no texto, apertamos a letra “n” minúscula, já o shift n retorna para a palavra anterior.

copy e paste no vi:

Para copiar algo, aperte “yy” 2x e colar aperte o “p”. Para copiarmos x linhas, ponha um número antes do yy

COMANDOS GERENCIAMENTO DE PROCESSOS

O comando Ps serve para listar os processos que estão ocorrendo em nosso linux:

As opções mais frequentes do ps são:

- a Mostra os processos em execução ligados a um terminal, de todos os usuários;
- -a Mostra os processos em execução ligados a um terminal, menos os processos de sessão;
- -e, -A Mostra todos os processos;
- -u Mostra a lista de processos incluindo o nome dos usuários donos dos processos e início das execuções, percentual de CPU utilizada, percentual de memória utilizada e terminal associado;
- -x Mostra a lista de processos, incluindo aqueles que não têm um terminal associado a ele. Útil para visualizar processos servidores (daemons);
- -f Mostra os processos em forma de árvore. Muito útil para identificarmos a relação de processo pai e filho entre os processos em execução;
- -H Mostra hierarquia dos processos em forma de árvore;

Neste exemplo o **ps** somente mostra os processos do usuário logado e ligados ao terminal:

```
$ ps
  PID TTY          TIME CMD
 1415 pts/0    00:00:00 ps
30019 pts/0    00:00:00 bash
```

Para mostrar todos os processos de todos os usuários ligados a um terminal:

```
$ ps a
  PID TTY      STAT   TIME COMMAND
 1628 pts/0    R+     0:00 ps a
 3297 tty1     Ss+    0:00 /sbin/agetty --noclear tty1 linux
27159 pts/0    T      0:00 sudo find / -iname backup.sh
27160 pts/0    T      0:00 find / -iname backup.sh
30019 pts/0    Ss     0:00 -bash
```

Repare como a opção “-a” é diferente do “a”:

```
$ ps -a
  PID TTY          TIME CMD
 1675 pts/0    00:00:00 ps
27159 pts/0    00:00:00 sudo
27160 pts/0    00:00:00 find
```

A opção “u” adiciona alguns atributos dos processos:

```
$ ps au
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root      3297  0.0  0.0 121336  1600 tty1     Ss+  ago15   0:00 /sbin/agetty --noclear tty1 linux
root      3298  0.0  0.1 120984  2064 ttyS0    Ss+  ago15   0:00 /sbin/agetty --keep-baud 115200,38400,9600
ec2-user  3414  0.0  0.1 164440  4032 pts/0    R+   18:38   0:00 ps au
root     27159  0.0  0.3 216984  6608 pts/0    T    17:46   0:00 sudo find / -iname backup.sh
root     27160  0.0  0.1 128308  3944 pts/0    T    17:46   0:00 find / -iname backup.sh
ec2-user 30019  0.0  0.2 127120  4348 pts/0    Ss   14:48   0:00 -bash
```

Para obter uma lista completa dos processos em execução, não só aqueles que estão conectados ao terminal, adicione a opção “x”:

```
$ ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.2 199452  4968 ?        Ss   ago15   9:23 /usr/lib/systemd/systemd --switched-root --
root         2  0.0  0.0      0     0 ?        S    ago15   0:00 [kthreadd]
( ... )
ec2-user 30018  0.0  0.2 152864  4384 ?        S    14:48   0:00 sshd: ec2-user@pts/0
ec2-user 30019  0.0  0.2 127120  4348 pts/0    Ss   14:48   0:00 -bash
postfix  30391  0.0  0.3  90536  6928 ?        S    18:06   0:00 pickup -l -t unix -u
```

Os processos cujo comando estão envoltos em chaves, como no destaque do **[ktheradd]**, indicam que eles foram retirados da memória RAM, e colocados na memória virtual em disco. Quando os processos estão na memória virtual em disco, são chamados de **sleeping**.

As opções “efH” mostram todos os processos, com a hierarquia deles em forma de árvore:

```
$ ps -efH
UID        PID  PPID  C STIME TTY          TIME CMD
root      3252     1  0 ago15 ?        00:00:00   /usr/sbin/sshd -D
root     29998  3252  0 14:48 ?        00:00:00     sshd: ec2-user [priv]
ec2-user 30018 29998  0 14:48 ?        00:00:00       sshd: ec2-user@pts/0
ec2-user 30019 30018  0 14:48 pts/0    00:00:00         -bash
ec2-user  4176 30019  0 18:43 pts/0    00:00:00           ps -efH
```

É possível brincar com os comandos, como, por exemplo, saber quais são os processos que mais consomem a CPU:

```
$ ps aux | sort -nrk 3,3 | head -n 5
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
nginx     3342  0.2  1.6 426976 34048 ?        Sl   ago15 133:04 amplify-agent
rpc       2729  0.0  0.1  73828  3276 ?        Ss   ago15   0:02 /sbin/rpcbind -w
root      9421  0.0  0.0      0     0 ?        I    set13   0:01 [kworker/u30:1]
root         9  0.0  0.0      0     0 ?        I    ago15   0:00 [rcu_bh]
```