Capítulo 2: Arquivamento e Compactação, Comandos Diversos, Pipe, Head, Tail, More, Less, Wc

Type	Leitura
Materials	Capítulo 2.pdf
Reviewed	☐

▼ Compactação de Arquivos

Arquivamento e Compactação

Arquivamento: Agrupar vários arquivos e/ou diretórios em um único arquivo.Tar = tape archive (arquivo de fita)

```
 zip [opcao] [nome_diretorio.zip] [nome do direotorio]

 zip -r diretorio01ediretorio02.zip diretorio01/ diretorio
```

```
tar [opções] [nome_do_arquivo_de_saida.tar.gz] [arquivo_ou
```

```
tar -czf arquivo.tar.gz /caminho/para/diretório
```

1. **gzip**:

gzip [opções] [arquivos]

- Utiliza o algoritmo de compressão Lempel-Ziv.
- Rápido e eficiente para compressão/descompressão.
- Menor taxa de compressão em comparação com `bzip2` e `xz`, mas é mais rápido.
- Extensão padrão dos arquivos compactados é `.gz`.
- Comumente utilizado para compactar arquivos individuais ou durante a transferência de dados na internet.
- Não suporta compactação de múltiplos arquivos/diretórios sem primeiro agrupá-los em um arquivo de arquivamento (`tar`).

1. **bzip2**:
   - Utiliza o algoritmo de compressão Burrows-Wheeler e Huffman.
   - Oferece uma taxa de compressão maior em comparação com `gzip`, mas é mais lento.
   - Extensão padrão dos arquivos compactados é `.bz2`.
   - É mais eficiente na compactação de arquivos de texto ou dados repetitivos.
   - Suporta compactação de múltiplos arquivos/diretórios sem a necessidade de agrupá-los em um arquivo de arquivamento.
2. **xz**:
   - Utiliza o algoritmo de compressão LZMA (Lempel-Ziv-Markov chain Algorithm).
   - Oferece uma das melhores taxas de compressão, mas é mais lento.
   - Extensão padrão dos arquivos compactados é `.xz`.
   - É comumente utilizado para compactar arquivos grandes ou distribuição de software.
   - Fornece uma excelente taxa de compressão para dados complexos ou arquivos binários.
3. **zip**:
   - Utiliza o algoritmo de compressão DEFLATE.

- É um formato de arquivo mais universalmente reconhecido e amplamente utilizado em sistemas Windows.
- Extensão padrão dos arquivos compactados é `.zip`.
- Suporta compactação de múltiplos arquivos/diretórios sem a necessidade de agrupá-los em um arquivo de arquivamento.
- Pode ser utilizado para criar arquivos de arquivamento que incluem estrutura de diretórios.

Usando o tar:

- Usa-se tar.gz ou tar.gz2
- tar [opções] [arquivo/diretorio.tar.gz] [arquivo/diretorio]

```
tar -cvf arquivo.tar arquivo1 arquivo2 diretório1

    tar -A: Concatena arquivos de backup existentes em um 
    tar -c, --create: empacotar arquivos e diretórios em u
    tar -d, --diff, --compare: Compara um arquivo de backu
    tar -t, --list: Lista o conteúdo de um arquivo de back
    tar -r, --append: Anexa arquivos a um arquivo de backu
    tar -u, --update: Atualiza um arquivo de backup existe
    tar -x, --extract, --get: Extrai o conteudo do arquivo


    - t (list os comandos) : tar -tzf ~/backup/arquivo01_1

-c: Cria um novo arquivo tar.
-z: Usa o gzip para compactar o arquivo.
-f: diz ao tar onde gravar o arquivo tar resultante ou de 
-C: muda o diretorio antes de arquivar o conteudo

-t: Lista o conteúdo do arquivo tar.
-f: Especifica o nome do arquivo tar.
gzip -r --recursive = Compacta arquivos em todos os diretó

 criar um arquivo de arquivamento e compactá-lo com gzip d
```

```
tar -czf ~/backup/diretorio01.tar.gz ~/diretorio01 && mv ~

Utilizando o comando tar, liste os arquivos compactados e,
arquivamento e a compactação feita na questão anterior.
tar -tf ~/backup/diretorio01.tar.gz
tar -xzf ~/backup/diretorio01.tar.gz

        tar -cjf = gzip2


    tar -cvf arquivo.tar arquivo1 arquivo2: Cria um arquiv
    tar -xvf arquivo.tar: Extrai o conteúdo do arquivo de
    tar -tvf arquivo.tar: Lista o conteúdo do arquivo de a
```

```
Compactar: gzip arquivo.txt
Descompactar: gzip -d arquivo.gz
Exibir conteúdo: zcat arquivo.gz

Compactar: bzip2 arquivo.txt
Descompactar: bzip2 -d arquivo.bz2
Exibir conteúdo: bzcat arquivo.bz2
bzip2recover: Recupera dados de um arquivo compactado co
m bzip2 danificado.

Compactar: xz arquivo.txt
Descompactar: xz -d arquivo.xz

Compactar: zip arquivo.zip arquivo.txt
Descompactar: zip -d arquivo.zip
Excluir: zip -x arquivo.zip
Recursivo: zip -r arquivo.zip


num - ajusta a taxa d compactação (gzip -9 texto.txt)

gzip -9 *.txt = Mantendo o
arquivo original e no nível máximo de compressão
```

gzip -a = Força o gzip a criar um arquivo no formato ASC II, ignorando bytes nulos.

gzip -c = deseja compactar um arquivo e ver a saída comp actada sem substituir o arquivo original

gzip -v = Modo verbose, exibe mensagens detalhadas de co mpactação/descompactação.ARQUIVOS QUE ESTAO SENDO COMPAC TADOS

gzip -d = Descompacta os arquivos compactados

gzip -f = Força a compactação

gzip -k = Mantém o arquivo original após a compactação (não o remove).

gzip -l --list = Lista informações sobre os arquivos com pactados sem descompactá-los.

gzip -L --license = Exibe a licença do software gzip .
gzip -n --no-name = Inclui o nome do arquivo original no cabeçalho do arquivo compactado.

gzip -N --name = Não inclui o nome do arquivo original n o cabeçalho do arquivo compactado.

gzip q --quiet = Suprime a maioria das mensagens de avi so e erro durante a execução do gzip.

gzip -r --recursive = Compacta arquivos em todos os dire tórios e subdiretórios de forma recursiva.

gzip -s = Reduz a memória usada pelo gzip durante a comp actação, mas pode aumentar o tempo de compactação.Bom co

```
m máquinas com poucos recursos
```

- Usado para o gzip, bzip2, xz, zip:
  - zip [opções] [arquivo-destino] [arquivos-origem]
  -

zip -r projeto.zip . -x '*.mp4'

▼ Expressões Regulares

Caracteres Coringas / Expressões Regulares

- **grep [comando] '[expressão]' [lugar]**
  - ***Ex:*** `grep '[^aeiou]' exemplo.txt`

1. `^`: Início de uma string
   - `grep '^cat' arquivo.txt` encontra linhas que começam com "cat" em "arquivo.txt".
2. `$`: Fim de uma string
   - `grep 'dog$' arquivo.txt` encontra linhas que terminam com "dog" em "arquivo.txt".
3. `.`: Qualquer caractere (exceto `\n` nova linha)
   - `grep 'c.t' arquivo.txt` encontra "cat", "cot", "cut" em "arquivo.txt".
4. `[ ]`: Conjunto explícito de caracteres para correspondência
   - `grep '[aeiou]' arquivo.txt` encontra qualquer linha com uma vogal em "arquivo.txt".

   grep -v '[0-9]' arquivo.txt
5. `*`: 0 ou mais da expressão anterior

- **Uso**: `grep 'ba*' arquivo.txt` encontra "b", "ba", "baa", etc., em "arquivo.txt".

6. `[^ ]`: Nenhum dos caracteres definidos
   - `grep '[^aeiou]' arquivo.txt` encontra caracteres que não são vogais em "arquivo.txt".
7. `\`: caractere especial.
   - `grep '\.' arquivo.txt` encontra linhas com o caractere '.' em "arquivo.txt".
8. `' '`: busca algo específico.
   - `grep 'r..f'` encontra palavra que comece com r e termina com f

9. **Asterisco `(*)`:**
   - O asterisco corresponde a zero ou mais caracteres em um nome de arquivo ou diretório.
   - Exemplo: Para listar todos os arquivos de texto em um diretório, você pode usar *.txt.
10. **Ponto de interrogação `(?)`:**
    - Unico caractere em um nome de arquivo ou diretório.
    - Exemplo: Para listar arquivos que tenham um único caractere no nome seguido por ".txt", você pode usar ?.txt. ou ls a?t*
11. **Colchetes `([])`:**
    - Especificado em uma posição.**Um caractere que comece de m [a-z] e vá até de 'a' a 'z'**
    - **Achar lista de opção todas as palavras que começam com 'x' mas não terminam com 'a, b, c,'**
    - Exemplo: Para listar arquivos que terminem com "1" ou "2", você pode usar *[12].
    - `ls m[^abc]` → Começam com a letra `m`.Seguidos por exatamente um caractere que não seja `a`, `b`, ou `c`.
12. **Chaves `({})` - Expansão de intervalo:**

- lista de sequência de caracteres.
- Exemplo: Para listar arquivos que correspondam a "arquivo1," "arquivo2," ou "arquivo3," você pode usar **arquivo{1,2,3}**
- `ls x{zd,ze}*`
  - Começam com a letra `x`.
  - Seguidos por `zd` ou `ze`.
  - Podem ter quaisquer caracteres adicionais após `zd` ou `ze`.

13. **Barra invertida ():**
    - A barra invertida é usada para escapar caracteres curinga, fazendo com que eles sejam tratados literalmente.
    - Exemplo: Se você quiser corresponder ao caractere de asterisco literalmente, use *.

COMANDOS

- `i`: Ignorar diferenças entre maiúsculas e minúsculas.
  - **Exemplo**: `grep -i 'padrão' arquivo.txt` encontra "padrão", "Padrão", "PADRÃO", etc.
- `v`: Inverter a correspondência (mostrar linhas que **não** contêm o padrão).
  - **Exemplo**: `grep -v 'padrão' arquivo.txt` exibe todas as linhas que não contêm "padrão".
- `r` **ou** `R`: Buscar recursivamente em diretórios.
  - **Exemplo**: `grep -r 'padrão' /diretorio` busca "padrão" em todos os arquivos dentro de `/diretorio`.
- `l`: Listar apenas os nomes dos arquivos que contêm o padrão.
  - **Exemplo**: `grep -l 'padrão' *.txt` mostra os arquivos `.txt` que contêm "padrão".
- `n`: Mostrar o número da linha onde o padrão ocorre.
  - **Exemplo**: `grep -n 'padrão' arquivo.txt` exibe "padrão" e o número da linha em que aparece.
- `c`: Contar o número de linhas que contêm o padrão.

    - **Exemplo:** `grep -c 'padrão' arquivo.txt` exibe o número de linhas que contêm "padrão".
- `w`: Correspondência de palavras inteiras (só encontra o padrão se estiver separado por espaços).
    - **Exemplo:** `grep -w 'padrão' arquivo.txt` encontra "padrão" como uma palavra inteira.
- `x`: Correspondência de linha inteira (só encontra se a linha inteira corresponder ao padrão).
    - **Exemplo:** `grep -x 'linha' arquivo.txt` encontra linhas que são exatamente "linha".
- `e`: Usar várias expressões regulares.
    - **Exemplo:** `grep -e 'padrão1' -e 'padrão2' arquivo.txt` encontra linhas que contêm "padrão1" ou "padrão2".
- `A [num]`: Mostrar [num] linhas após a linha correspondente.
    - **Exemplo:** `grep -A 3 'padrão' arquivo.txt` mostra 3 linhas após cada linha que contém "padrão".
- `B [num]`: Mostrar [num] linhas antes da linha correspondente.
    - **Exemplo:** `grep -B 2 'padrão' arquivo.txt` mostra 2 linhas antes de cada linha que contém "padrão".
- `C [num]`: Mostrar [num] linhas antes e depois da linha correspondente.
    - **Exemplo:** `grep -C 4 'padrão' arquivo.txt` mostra 4 linhas antes e 4 linhas depois de cada linha que contém "padrão".

Aqui estão alguns exemplos de como usar caracteres curinga em comandos no terminal:

- `ls *.txt ou ls a*r`: Lista todos os arquivos de texto no diretório atual.
- `rm arquivo?.txt ou ls m??`: Remove arquivos com um único caractere antes de ".txt."

- `mv arquivo[12] destino/`: Move arquivos que terminam com "1" ou "2" para o diretório "destino."
- `cp arquivo{1,2,3} copias/: Copia "arquivo1,"` "arquivo2," e "arquivo3" para o diretório "copias."
- `cat \*.*:` Exibe o conteúdo de todos os arquivos com uma extensão no diretório atual, ignorando o diretório atual "." e o diretório pai "..".

▼ Contagem | Corte | Comparação de palavras | Enumeração

WC

- contar linhas, palavras e caracteres em um arquivo ou na entrada padrão.

`wc`: Abreviação de "word count", é uma ferramenta de utilidade para contar:

- Linhas (`l`)
- Palavras (`w`)
- Caracteres (`m` ou `c`)
- Bytes (`c`)

CUT

`cut [opções] [arquivo]`

- `f`: Especifica os campos a serem extraídos. Você deve usar em conjunto com a opção `d` para definir o delimitador.
  - Exemplo: `cut -f1,3 arquivo.txt` extrai o 1º e 3º campos do arquivo.
- `d`: Define o delimitador de campo (o caractere que separa os campos).
  - Exemplo: `cut -d',' -f2 arquivo.csv` usa a vírgula como delimitador e extrai o 2º campo.
- `c`: Especifica as posições de caracteres a serem extraídos.

    - Exemplo: `cut -c1-5 arquivo.txt` extrai os caracteres da posição 1 a 5 de cada linha.
- `s`: Supressão de linhas que não contêm o delimitador.
    - Exemplo: `cut -d',' -f1 -s arquivo.txt` só mostra linhas que contêm uma vírgula.

CORTANDO CAMPOS

`cut -d ":" -f 1 /etc/passwd` - pega o primeiro campo

`cut -d ":" -f 1, 2 /etc/passwd` - pega o primeiro e segundo campo

`cut -b 1-4 /etc/passwd` - pega os bytes de 1 ao 4, pega todos os espaços

`cut -c 1-4 /etc/passwd` - pega os bytes de 1 ao 4, porém sem contar os espaços

COMPARAR ARQUIVOS

`cmp arquivo1 arquivo2` - compara os 2 arquivos

`diff` - é um cmp em mais legível

`diff -u , -r` - comparar as pastas, com um símbolo de '+' e '-'

ENUMERAÇÃO DE LINHAS

`head -c 10 passwd` → **10 primeiros bytes**

`cat /etc/passwd | sort -t ":" +2 -n | head -n 1` → **10 primeiros bytes**

`nl /etc/passwd` → **enumerada, de maneira bonita, as linhas**

`nl -f a -i 2 /etc/passwd` → **vai de 2 em 2**

`ln -f a -i 3 -v 2 /etc/passwd` → **começar por 0 a enumerar**

MORE, LESS e SORT

exemplo: `cat /etc/ssh/ssh_config | more`

exemplo: `cat /etc/ssh/ssh_config | less` **→ consigo rolar para cima e baixo.Para sair pressionar o 'q' Eu também consigo procurar coisas dentro do arquivo.Basta digitar o '/'.**

`sort arquivo2.txt` **→ ordena os arquivos, primeiro os números e depois os nomes.**

`cat arquivo2.txt | sort -r` **→ ordena os nomes para depois os arquivos**

`cat arquivo2.txt | sort -n` **→ ordena por ordem númera.Se você não colocar o '-n', ele vai ordenar como se fosse string, pois ele entende como string.**

`cat arquivo2.txt | sort -c` **→ diz se tá desordenado ou não**

`cat arquivo2.txt | sort +1` **→ tá ordenando pela segunda coluna.Nesse caso se eu colocar Allyson Augusto, ele ordenará o 'Augusto', em ordem alfabética**

`cat arquivo2.txt | sort +1 -t ''F''` **→ Quando ele ver a letra ''F'', ele vai ver que, tudo que está antes da letra ''F'' é a coluna 0 e posterior é coluna 1**

`cat arquivo2.txt | sort -k 1` **→ especificar que a primeira coluna (campo) deve ser usada como chave para a ordenação**

▼ Filtragem de palavras

FILTRANDO E BUSCANDO INFORMAÇÕES

PROCURANDO ARQUIVOS E PASTAS

- `du -hs` mostrará o tamanho total
- `find / -name "arquivo.txt"`
  - Procura por arquivos e diretórios chamados "arquivo.txt" a partir do diretório raiz `/`.
- `find . -name nome_do_arquivo`

    - Procura por arquivos e diretórios com o nome especificado a partir do diretório atual (`.`).
- `find . -type d -name mc`
    - Encontra apenas diretórios chamados "mc" a partir do diretório atual.
- `find . -type f -name mc`
    - Encontra apenas arquivos chamados "mc" a partir do diretório atual.
- `find /usr -maxdepth 2 -type f -name mc`
    - Procura por arquivos chamados "mc" dentro do diretório `/usr`, mas apenas até dois níveis de subdiretórios.
- `find . -mtime -1`
    - Lista arquivos que foram modificados no último dia.
- `find . -ctime -1`
    - Lista arquivos que foram criados no último dia.
- `find . -atime -1`
    - Lista arquivos que foram acessados no último dia.
- `find /usr -size +1000`
    - Lista arquivos dentro do diretório `/usr` que têm mais de 1000 bytes.
- `free --kilo ou --mega ou --kibi --mebi --gibi --giga` (memória em Kbytes ou Megabytes) → porque (quilo, mega, giga, etc.) usados no sistema métrico são baseados em potências de 10, enquanto as unidades de armazenamento de dados em computação são baseadas em potências de 2.
- `free --mega -s 1` → me traz a memória utilizada a cada 1s

PROCURANDO TEXTOS DENTRO DE ARQUIVOS

1. `grep ‘r.d’ red.txt`: Imprimirá linhas do arquivo 'red.txt' que contenham um 'r' seguido de qualquer caractere e depois 'd'.
2. `grep 'root' /etc/passwd`: Procurará por linhas no arquivo '/etc/passwd' que contenham a palavra 'root'.
3. `grep -v 'www-data' /etc/passwd`: Imprimirá todas as linhas do arquivo '/etc/passwd' exceto aquelas que contenham 'www-data'.
4. `grep -f /tmp/`: Procurará padrões contidos nos arquivos listados em '/tmp/'.
5. `grep -i 'WWW-DaTa' /etc/passwd`: Procurará por 'WWW-DaTa' no arquivo '/etc/passwd' sem diferenciação entre maiúsculas e minúsculas.
6. `grep -iE '^www-data.*nologin$'`: Procurará no arquivo '/etc/passwd' por linhas que começam com 'www-data' e terminam com 'nologin'.
7. `grep -iF '.*' /etc/passwd`: Imprimirá linhas do arquivo '/etc/passwd' que contenham pelo menos um caractere.
8. `grep -ri 'Foca02' .`: Procurará recursivamente por 'Foca02' nos arquivos e diretórios atuais, ignorando diferenças de maiúsculas e minúsculas.
9. `grep -rih`: Procurará recursivamente, ignorando maiúsculas e minúsculas, mas não imprimirá os nomes dos arquivos correspondentes.
10. `grep -ril 'Foca02' .`: Listará os arquivos que contêm 'Foca02', mas não imprimirá o conteúdo dos arquivos.

- `head passwd` → **mostra por padrão as 10 primeiras linhas**
- `head -n 3 passwd` → **3 primeiras linhas**
- `cat passwd | head -n 3`
- `echo www-data > /tmp/expressao` → **cria algo e já coloca em um lugar**

▼ Comandos Diversos (Date, Time, Dmesk, Tail e Head)

COMANDOS DIVERSO

DATE

- `sudo date --set="2024-08-02 22:30:00”`
- `sudo date 101007452020` (mes, dia, hora, minuto e ano) e para salvar na CMOS do sistema, utiliza-se o **hwclock --systohc**
- `date + "%d-%m-%Y %T"`
- `date +%R` → (formato de 24 horas)
- `date +"%d %Y %j"` → Quantos dias falta pra acabar o ano
- `df -h e df-Th` → Mostra as partições e seus respectivos sistemas de arquivos
- `df -aTh -t arquivo_especifico` → Busca uma especificação nas partições
- `ln -s minha_pasta meu_link` → Link simbólico que, qualquer operação realizada dentro de "meu_link", afetará o conteúdo de "PASTA".Ou seja, ln -s /usr/bin bin-usr. Toda vez que eu entrar no caminho /usr/bin ele vai para bin-usr

TIME e UPTIME

`uptime` → tempo de atividade da máquina desde o último boot

`time ls` → veja o tempo de execução do comando

DMESG, TALK e MESG

`dmesg` → fornece informações sobre o hardware, drivers de dispositivos e eventos do kernel do sistema

`dmesg -t | grep enp0s3`

`dmesg -x` → ***prioridade das mensagens em texto legível***

`dmesg -T` → ***legível***

`dmesg -c` → ***limpa as mensagens do buffer do kernel para esperar as outras***

`mesg` →***definir se você permite ou não que outros usuários enviem mensagens de bate-papo para você.Se digitar 'mesg y' ele ativa.Para desativar, digite 'mesg n'***

`talk` → ***mandando mensagem em tempo real***

ECHO

`echo` → ***mensagem na tela***

`echo -n "teste no linux admin"` → ***não faz quebra de linha***

`echo -e "Teste do Linux" -` → ***habilita os caracteres especiais***

DESLIGAR A MÁQUINA

`sync` - *gravar os buffers do kernel no disco, porque ao invés de esperar 20s , ele faz forçado*

`uname -a, -r, -n` -- *nome da máquina e suas especificações*

`echo b>/proc/sysrq-trigger` - *reiniciar para emergencia*

`halt - desligar`

`echo o>/proc/sysrq-trigger` -→ *desligar a máquina forçada, caso esteja com problemas de emergência*

`shutdown -h 09:40` -→ *agendamento para desligar*

`wc` - *retorna palavras, bytes e linhas de um arquivo (ex: wc /etc/passwd)*

`seq` - *sequência de números (ex: seq 10, seq 2 2 10, seq 2 10)*

TOUCH

`touch -t 10120815 /tmp/arquivo` → ***modifica a hora e data do arquivo0***

`touch -a -t 10120815 /tmp/arquivo` → ***modifica o acesso do arquivo***

ERROS:

TAIL , HEAD e LESS

```
head arquivos_com_a.txt   # Exibe as primeiras 10 linhas

less arquivos_com_a.txt   # Rolar para cima e para baixo n
```

```
tail arquivos_com_a.txt   # Exibe as últimas 10 linhas
tail -f /var/log/auth.log → #pega as última linhas do arqu
tail /etc/passwd --> #visualiza as 10 ultimas linhas
tail -n 4 /etc/passwd --> #visualiza as 4 ultimas linhas
tail -f /var/log/auth.log →
```

```
grep '^[[:space:]]*A' palavras.txt
```

```
grep -E "[0-9]{2}/[0-9]{2}/[0-9]{4}" datas.txt
```

```
grep "[?$]" perguntas.txt
```

```
grep -E '[a-z]{6}' dicionario.txt
```

```
grep -E '([0-9]{1,3}\.){3}[0-9]{1,3}' logs.txt
```

```
grep -E 'http[s]?://[a-zA-Z0-9./?=]+' links.txt
```

```
grep -E '[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,7}'
```