APOSTILA PARA INICIANTES GNU/LINUX

(comandos básicos, diretórios GNU/Linux e instalação servidor Web Apache 2)

Ayrton Pereira

Setembro 2022

Índice

Considerações iniciais

Antes de mais, quero saudar e agradecer aos leitores que vão ler ou apenas consultar determinados assuntos existentes nesta Apostila para iniciantes GNU/Linux que aborda sobre diversos assuntos de forma introdutória e resumida como: sistemas operativos no geral, GNU/Linux e licenças GNU GPL, comandos básicos de terminal para iniciantes, sistema de diretórios GNU/Linux e um pequeno bónus que é a instalação do servidor de aplicação 'Web', nomeadamente o Apache2. A criação desta apostila deu-se devido à vontade de facilitar-me consultar e memorizar os conteúdos aqui escritos e também querer transmitir o conhecimento adquirido por mim mediante tutoriais, pequenos cursos e leituras sobre este conteúdo. Deste modo estaria a ajudar iniciantes que estejam a iniciar no mundo de sistemas operativos GNU/Linux e a mim mesmo.

Espero, que o conteúdo nesta apostila ajude os utilizadores de sistema GNU/Linux. Agradeceria que em caso da vontade de utilizar a informação aqui presente, que fosse respeitada os direitos autorais e mencionar que proveio desta apostila. Em caso de algum desacordo sobre algo que foi aqui escrito ou vontade de acrescentar algo nesta apostila poderá entrar em contacto comigo para o endereço eletrónico endereçado no rodapé desta apostila ou pelo meios de rede sociais presentes também no rodapé,

1. Sistema operativo

Um sistema operativo consiste num conjunto de programas que atuam como intermediário entre o 'hardware' e 'software', ou seja, é um conjunto de programas responsáveis pela gestão dos recursos, processadores, dispositivos de entrada e saída de dados da máquina e os seus respetivos periféricos. Com isso, o sistema operativo tem certas funções básicas, como:

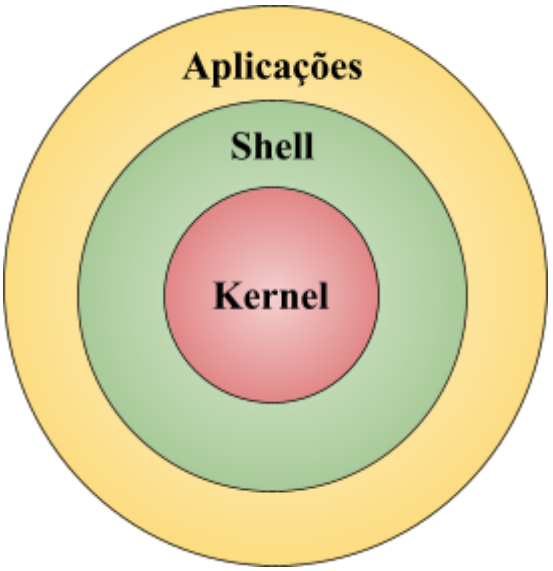
- Interpretar os comandos dos utilizadores;
- Controlar os periféricos;
- Organizar arquivos do disco.

Este permite a comunicação entre diversos elementos computacionais, e comunica-se com os utilizadores da máquina, hardware (parte física computacional), diversos tipos de 'software' ('software' de aplicativos, 'softwares' utilitários, compiladores, entre outros…), operadores computacionais (responsáveis por controlar o sistema operativo, normalmente em máquinas de grande porte, como funções de controlo de discos, fitas, impressora, etc.), programadores de aplicação, programadores de sistema, administradores de sistema.

1.1. Composição de um sistema computacional

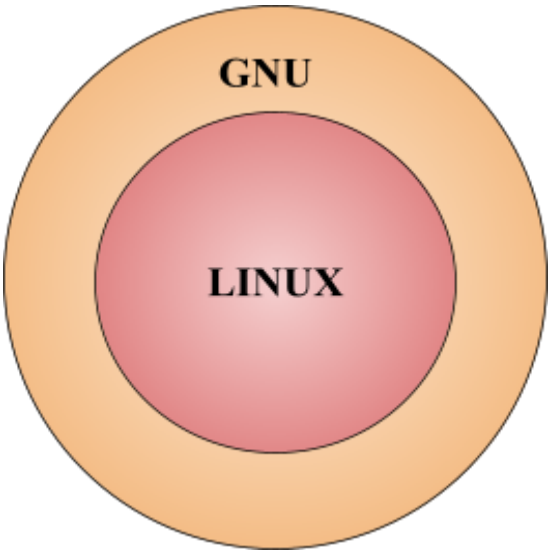
Um sistema computacional é composto por três partes, nomeadamente:

- Kernel (núcleo);
- 'Shell';
- Aplicações ou 'softwares'.



2. GNU/LINUX

O GNU é um acrónimo recursivo para **GNU is Not Unix** criada por Richard Stallman. Este é um projeto apoiado pela organização sem fins lucrativos **FSF (Free Software Foundation)** que disponibiliza uma coleção de diversos programas ('softwares') livres desde aplicações, bibliotecas, diversas ferramentas e até jogos. O termo 'software' livre significa que os utilizadores têm a liberdade de executar, copiar, distribuir, estudar, alterar e melhorar o 'software'. O seu intuito principal é de ser um sistema compatível com Unix que seria 100% livre.



Geralmente este projeto é usado em conjunto com o Kernel Linux, tornando-o num sistema operativo denominado por GNU/Linux.

2.1. Distribuições-base do GNU/LINUX

As distribuições-base são usadas por diversas entidades empresariais ou não empresariais, para desenvolver outras distribuições (distros GNU/Linux) utilizadas e testadas pelas comunidades de utilizadores de sistemas operativos GNU/LINUX ou apenas utilizadas no ambiente corporativo. As distribuições-base são:

- Slackware;
- Debian;
- RedHat;
- Raiz.

2.2. Licença GNU GPL

A licença GNU GPL (GNU General Public Licence) é uma designação da licença para o 'software' livre que foi idealizada por Richard Stallman no âmbito do seu projeto GNU. Esta é uma licença com mais uso por parte dos diversos projeto de 'software' livre, na maioria devido a sua adoção para o projeto GNU e o sistema operativo GNU/Linux. A licença GNU GPL garante 4 liberdades:

— A liberdade de executar o programa (executar o programas para diversos fins);
— A liberdade de estudar (podemos estudar, analisar o código-fonte);
— A liberdade de distribuir/redistribuir;
— A liberdade de modificar (modificar para os propósitos dos utilizadores);

2.3. Terminal GNU/Linux

O prompt do Linux é um emulador de terminal (pseudoterminal) onde é executado diversos comandos. Por baixo do terminal do GNU/Linux temos um interpretador de comandos, que neste caso será o 'Bash', mas, também existem muitos outros, porém o mais usado é o'Bash'. O terminal do GNU/Linux é case sensitive, ou seja, as letras maiúsculas e minúsculas são dadas em conta.

2.3.1. Comandos básicos do terminal

Os comandos no terminal são externos, ou seja, todos os comandos são programas executáveis que estão localizados numa pasta especifica que é o **'/bin'**. A partir do parâmetro `--help` de cada comando é possível saber quais os parâmetros que o comando tem, para um uso mais completo do comando,

Sintaxe:

```
comando  --help
```

Exemplo:

```
cd --help
```

Alguns comandos poderão não ser reconhecidos pelo terminal em algumas distribuições, nestes casos poderá haver a necessidade de instalar o comando para um uso posterior no sistema.

2.3.1.1. Comando man

O comando `-man` permite consultar o manual de um determinado comando. Utilizando o nome do comando a querer consultar como parâmetro do comando **man**.

Sintaxe:

```
man comando
```

Exemplo:

```
man ls
```

2.3.1.2. Comando ls

O comando -ls lista os arquivos de um determinado diretório.

Sintaxe:

```
ls <diretório_a_listar>
```

Exemplo:

```
ls ~
```

Podemos usar este comando com diversos parâmetros como:

ls -l ou ls --long : permite listar os arquivos e as suas propriedades como a data de criação, o tamanho, o utilizador, o grupo e tipo de arquivo.

Sintaxe do comando com o parâmetro abreviado:

```
ls -l <diretório_a_listar>
```

Sintaxe do comando com o parâmetro não abreviado:

```
ls -l <diretório_a_listar>
```

s -l -h : permite listar os arquivos e as suas propriedades de uma maneira mais legível ao homem.

Sintaxe:

```
ls -l -h <diretório_a_listar>
```

ls -a : permite listar todos os arquivos incluindo os arquivos ocultos existentes nesse diretório, exibindo os arquivos/diretórios antecedido com '.'.

Sintaxe:

```
ls -a <diretório_a_listar>
```

ls -al : permite listar todos os arquivos incluindo os ocultos e também mostrar as suas propriedades.

Sintaxe:

```
ls -al <diretório_a_listar>
```

ls -R : permite listar exibindo o conteúdo nos repositórios onde a listagem é desejada. Esta listagem é denominada listagem recursiva.

Sintaxe:

```
ls -R <diretório_a_listar>
```

2.3.1.3. Comando cd (change directory)

O comando `-cd` permite entrar e sair de pastas/diretórios. Quando utilizado com os sinais:

cd ~ : permite retornar a pasta de utilizador.

cd .. : permite voltar ao diretório pai, ou seja, ao diretório anterior ao atual.

Sintaxe:

```
cd <diretório_a_listar>
```

2.3.1.4. Comando lscpu

O comando `-lscpu` mostra as propriedades do cpu da máquina.

Sintaxe:

```
lscpu
```

2.3.1.5. Comando pwd (print working directory)

O comando `-pwd` permite visualizar o caminho até ao diretório atual.

Sintaxe:

```
pwd
```

2.3.1.6. Comando mkdir (make directory)

O comando `-mkdir` permite criar um ou mais diretórios, durante ao uso do comando.

Sintaxe:

```
mkdir <nome_do_diretório_a_ser_criado>
```

Exemplo (criação de apenas um diretório):

```
mkdir ./pasta1
```

Exemplo (criação de dois diretórios):

```
mkdir ./pasta1 ./pasta2
```

mkdir -p ou mkdir --parents: o comando `-mkdir` com o parâmetro `--parents` ou abreviado `-p` permite criar pastas/diretórios com subpastas/subdiretórios.

Sintaxe do comando com o parâmetro abreviado:

```
mkdir -p <nome_da_pasta/nome_da_subpasta>
```

Sintaxe do comando com o parâmetro não abreviado:

```
mkdir --parents <nome_da_pasta/nome_da_subpasta>
```

Exemplo:

```
mkdir -p pasta1/pasta2
```

2.3.1.7. Comando touch

O comando `-touch` permite criar arquivos sem um formato especifico ou com um formato especifico dependendo da extensão.

Sintaxe:

```
touch <nome_arquivo.extensão>
```

Exemplo:

```
touch arquivo.pdf
```

2.3.1.8. Comando history

O comando `-history` permite visualizar o histórico dos comandos usados no terminal, estes registos de comandos serão exibidos numericamente consoante a ordem de uso no terminal.

Sintaxe:

```
history
```

2.3.1.9. Comando cat

O comando `-cat` é um comando de manipulação de arquivos. Este comando permite visualizar o conteúdo do arquivo na tela do terminal, sem a possibilidade de o modificar.

Sintaxe:

```
cat <nome_do_arquivo.extensão>
```

Exemplo:

```
cat arquivo.pdf
```

2.3.1.10. Comando rmdir

O comando `-rmdir` permite excluir diretórios vazios.

Sintaxe:

```
rmdir <caminho_até_ao_diretorio_vazio_a_excluir>
```

Exemplo

```
rmdir ./diretório_vazio_para_excluir
```

2.3.1.11. Comando rm

O comando `-rm` permite excluir arquivos em diretórios.

Sintaxe:

```
rm <caminho_até_ao_diretorio_ou_arquivo>
```

Podemos usar este comando com diversos parâmetros como:

rm -rf: o uso deste comando com parâmetro **-rf** força a exclusão de tudo que estiver no diretório, incluindo o próprio diretório ou qualquer arquivo.

Sintaxe:

```
rm -rf <caminho_até_ao_diretorio/arquivo
```

rm -rf -i: o uso deste comando com parâmetro **-rf** e **-i** força a exclusão, mas antes questiona se realmente quer excluir.

Sintaxe:

```
rm -rf -i <caminho_até_ao_diretorio/arquivo>
```

2.3.1.12. Comando less

O comando `-less` é um paginador, utilizado para realizar consultas ao arquivo. Este comando permite realizar buscas no arquivo.

Sintaxe:

```
less <nome_do_arquivo.extensão>
```

Exemplo:

```
less arquivo.pdf
```

2.3.1.13. Comando tac

O comando `-tac` tem o mesmo comportamento que o comando `-cat`, de permitir visualizar o conteúdo do arquivo na tela do terminal. Porém, com diferença que o conteúdo do documento é exibido da última linha para a primeira (inversamente).

Sintaxe:

```
tac <nome_do _arquivo.extensão>
```

Exemplo:

```
tac arquivo.pdf
```

2.3.1.14. Comando cp

O comando `-cp` permite copiar diretórios ou arquivos. Ao usar o comando devemos especificar primeiramente o que queremos copiar e o destino.

Sintaxe:

```
cp <caminho_para_arquivo_original> <caminho_para_destino_cópia_do_arquivo>
```

Exemplo:

```
cp ./arquivo.pdf ./diretorioExemplo/arquivoCopiado.pdf
```

2.3.1.15. Comando mv

O comando `-mv` permite alterar o nome de um arquivo ou diretório e também mudar o local onde um determinado arquivo ou diretório está armazenado.

Sintaxe:

```
mv <caminho_origem_do_arquivo> <caminho_destino_do_arquivo>
```

Exemplo alterando o nome de um arquivo/diretório:

```
mv ./arquivo.pdf ./arquivo_novo_nome.pdf
```

Exemplo alterando o local de armazenamento do arquivo/diretório:

```
mv ./arquivo.pdf ./diretório_exemplo/arquivo.pdf
```

2.3.1.16. Comando whoami

O comando `-whoami` é um comando que serve para perguntar ao sistema quem é o utilizador.

Sintaxe:

```
whoami
```

2.3.1.17. Comando su

O comando `-su` permite fazer o login no terminal como "super user ou substitute user" com privilégios de super usuário ou trocar para outro utilizador existente na máquina caso este seja definido após o comando.

Sintaxe para efetuar login como super usuário:

```
su
```

Sintaxe para efetuar 'login' com outro utilizador existente no sistema:

```
su   nomeUsuario
```

Exemplo para efetuar 'login' com outro utilizador existente no sistema:

```
su usuarioB
```

2.3.1.18. Comando echo

O comando `-echo` serve para imprimir um determinado texto na tela.

Sintaxe:

```
echo "Texto"
```

Exemplo:

```
echo "gato"
```

2.3.1.19. Comando wc

Comando `-wc` retorna o total de linhas, palavras e caracteres, incluindo os espaços em branco de um determinado documento.

Sintaxe:

```
wc <nome_arquivo.extensão>
```

Exemplo:

```
wc arquivo.pdf
```

Este comando pode ser utilizado com determinados parâmetros que permitem retornar apenas o total de linhas, palavras ou caracteres, incluindo os espaços em branco. Nomeadamente:

wc -l : o uso deste comando com o parâmetro **-l** permite retornar o número total de linhas de um documento.

Sintaxe:

```
wc -l <nome_arquivo.extensão>
```

wc -w : o uso deste comando com o parâmetro **-w** permite retornar o número total de palavras de um documento.

Sintaxe:

```
wc -w <nome_arquivo.extensão>
```

wc -c : o uso deste comando com o parâmetro **-c** permite retornar o número total de caracteres, incluindo os espaços em branco, de um documento.

Sintaxe:

```
wc -c <nome_arquivo.extensão>
```

2.3.1.20. Comando date

O comando `-date` permite visualizar a data atual e também permite alterar a data atual do sistema.

Sintaxe:

```
date
```

2.3.1.21. Comando uname

O comando `-uname` permite-nos saber a arquitetura do sistema operativo.

Sintaxe:

```
uname
```

Este comando pode ser utilizado com os seguintes parâmetros que nos permite saber, por exemplo, a arquitetura, o kernel entre outros, nomeadamente:

uname -m: o uso deste comando com o parâmetro **-m** permite saber a arquitetura do sistema operativo.

Sintaxe:

```
uname -m
```

uname -v: o uso deste comando com o parâmetro **-v** permite saber a versão do kernel.

Sintaxe:

```
uname -v
```

uname -a: o uso deste comando com o parâmetro **-a** permite obter informação sobre kernel, sistema operativo, nome do computador, entre outras informações.

Sintaxe:

```
uname -a
```

2.3.1.22. Comando chmod

O comando `-chmod` permite configurar as permissões dos diversos arquivos (permissões no modo octal) para poder dar determinadas permissões a um arquivo ou diretório devemos ter permissões de utilizador root, ou ser donos do arquivo, ou diretório. As permissões baseadas em **chmod** são:

- r (read) para leitura com peso de 4;
- w (write) para escrita com o peso de 2;
- x (execute) para a execução com o peso de 1

Ao configurar as permissões temos que ter em conta o utilizador dono do arquivo, o grupo a que o dono do arquivo pertence e outros utilizadores e grupos do sistema. Para que um arquivo tenha permissão de leitura, escrita e execução (permissão total) temos que somar os pesos todos que será 7; se quisermos que apenas tenha a permissão de leitura e escrita terá que se somar os seus pesos e atribuir 6; assim, respetivamente para outro tipo de permissões.

Sintaxe:

```
chmod
<peso_para_dono_arquivo><peso_para_grupo_utilizador><peso_para_out
ros_utilizadores_grupos> <caminho_até_ao_arquivo>
```

Exemplo:

```
chmod 777 arquivoExemplo.pdf
```

Quem apenas poderá alterar as permissões é o dono do arquivo e o utilizador root. O utilizador root poderá efetuar qualquer coisa. Quando configuramos as permissões como root, automaticamente o dono do arquivo será o utilizador root.

2.3.1.23. Comando chown

O comando `-chown` permite alterar o dono de um arquivo ou de um diretório.

Sintaxe:

```
chown <novo_dono> <caminho_até_ao_arquivo>
```

2.3.1.24. Comando chgrp

O comando `-chgrp` permite alterar o grupo a qual o arquivo pertence.

Sintaxe:

```
chgrp <nome_do_grupo> <caminho_até_ao_arquivo>
```

2.3.1.25. Comando useradd

O comando `-useradd` é utilizado para adicionar novos utilizadores ao sistema operativo. Neste comando devemos passar como parâmetro o grupo a que o utilizador criado pertence através do parâmetro **-g**, o seu shell através do parâmetro **-s**, o seu diretório através do parâmetro **-d** e comentário através do parâmetro **-c**, o parâmetro **-m** para o nome de utilizador.

Sintaxe:

```
useradd -g <nome_do_grupo> -s <caminho_até_ao_interpretador_de_linha_de_comando> -d <diretório_do_utilizador> -c <comentário> -m <nome_do_utilizador>
```

2.3.1.26. Comando passwd

O comando `-passwd` é utilizado para definir uma nova senha de um utilizador do sistema. Quando tem a sessão iniciada como root terá privilégio para alterar senha de qualquer utilizador do sistema, quando for utilizador normal do sistema poderá apenas alterar a sua senha.

Sintaxe quando é root:

```
passwd <nome_do_utilizador>
```

Sintaxe quando é utilizador normal:

```
passwd
```

2.3.1.27. Comando gpasswd

O comando `-gpasswd` permite fazer a gestão dos grupos do sistema e os seus utilizadores. Podemos usar este comando com diversos parâmetros. Quando usamos este comando com o parâmetro **-a** irá permitir-nos associar um determinado utilizador a um determinando grupo.

Sintaxe:

```
gpasswd -a ”nome do utilizador” “nome do grupo”
```

2.3.1.28. Comando userdel

O comando `-userdel` permite excluir utilizadores, mas não exclui os arquivos pessoais do utilizador.

Sintaxe:

```
userdel "nome do utilizador"
```

Este comando pode ser utilizado com o parâmetro **-r** permitirá excluir o utilizador e também os seus arquivos pessoais.

Sintaxe:

```
userdel -r "nome do utilizador"
```

2.3.1.29. Comando lspci

O comando `-lspci` permite listar os dispositivos conectados a nossa máquina.

Sintaxe:

```
lspci
```

2.3.1.30. Comando lsusb

O comando `-lsusb` permite listar os dispositivos conectados às portas usb da máquina.

Sintaxe:

```
lsusb
```

2.3.1.31. Comando lsmod

O comando `-lsmod` permite listar os módulos do Linux, ou seja, listar os drivers.

Sintaxe:

```
lsmod
```

2.3.1.32. Comando top

O comando `-top` permite visualizar os processos do sistema operativo.

Sintaxe:

```
top
```

2.3.1.33. Comando tree

O comando `-tree` permite visualizar o sistema de arquivos em forma de árvore. Irá mostrar a estrutura em árvore dos documentos dependendo do diretório.

Sintaxe:

```
tree <caminho_até_diretório>
```

2.3.1.34. Comando free

O comando `-free` permite mostrar informações sobre a memória RAM e também da memória SWAP. Ao usar o comando `-free` podemos especificar os valores em unidades desejadas a serem exibidas, para isso devemos usar o comando `-free` com os seguintes parâmetros: **-k** para aparecer em unidade kb, **-m** para aparecer em unidade mb (megabyte), **-g** para aparecer em unidade gb e para aparecerem as letras das unidades usamos o parâmetro **-h**.

Sintaxe:

```
free -<parâmetro>
```

2.3.1.35. Comando df

O comando `-df` permite mostrar as informações sobre o disco e também o sistema de arquivo. Quando usamos o comando com o parâmetro **-h** permite-nos visualizar as informações mais detalhadas.

Sintaxe:

```
df -<parâmetro>
```

2.3.1.36. Comandos gestores de pacotes

Os comandos de gestão dos pacotes do sistema, são responsáveis pela instalação de 'softwares', desinstalação de 'softwares' entre outras ações como, por exemplo, atualização dos mesmos através da linha de comando. Existem diversos gestores de pacotes (package manager), porém abordarei do `-dpkg` e `-apt` que são os mais usados nas distribuições baseadas em Debian e no próprio Debian.

2.3.1.36.1. Gestor de pacotes apt

O gestor de pacote apt permitir-nos realizar downloads de 'softwares' (repositórios) diretamente dá 'internet' pelo terminal. Para usarmos este gestor, devemos usar o comando `-apt` com privilégios de administrador, em conjunto com certos parâmetros conforme a necessidade. Abaixo estão alguns parâmetros básicos que podem ser usados no dia a dia com o apt:

Sintaxe:

```
sudo apt parâmetro --<opção_do_parâmetro_se_tiver>
```

apt install: o comando `-apt` com parâmetro **install** permite instalar um único pacote ou vários pacotes.

Sintaxe instalação de um único pacote:

```
sudo apt install nome_do_pacote
```

Sintaxe instalação de mais de um pacote:

```
sudo apt install nome_do_pacote1 nome_do_pacote2  ...
```

apt remove: o comando `-apt` com parâmetro **remove** permite remover pacotes que foram instalados. A execução deste comando pode ser utilizado para remover um único pacote ou vários pacotes que já foram instalados.

Sintaxe para remover um único pacote:

```
sudo apt remove nome_do_pacote
```

Sintaxe para remover vários pacotes:

```
sudo apt remove nome_do_pacote1 nome_do_pacote2 ...
```

apt list: o comando `-apt` com parâmetro **list** permite listar os pacotes disponíveis, instalados e os pacotes que são atualizáveis. Para podermos também visualizar os pacotes instalados devemos usar a opção **--installed**, para ver os pacotes que podem ser atualizáveis devemos usar a opção **--upgradeable**.

Sintaxe para visualizar os pacotes disponíveis:

```
sudo apt list
```

Sintaxe para visualizar os pacotes instalados:

```
sudo apt list --installed
```

Sintaxe para visualizar os pacotes atualizáveis:

```
sudo apt list --upgradeable
```

apt update: o comando `-apt` com o parâmetro **update** permite atualizar o banco de dados onde contém os registos dos pacotes disponíveis no sistema, a partir das alterações mais recentes dos repositórios do APT.

Sintaxe:

```
sudo apt update
```

apt upgrade: o comando `-apt` com o parâmetro **upgrade** permite realizar atualizações para versões mais recentes dos pacotes instalados no sistema, ou apenas, atualizar um único pacote instalado, quando este exija a remoção de um determinado pacote instalado para a sua atualização.

Sintaxe para atualizar pacotes para versões mais recentes:

```
sudo apt upgrade
```

Sintaxe para atualizar um único pacotes para versões mais recentes:

```
sudo apt upgrade nome_do_pacote
```

apt autoremove: O comando `-apt` com o parâmetro **autoremove** permite remover dependências (dependências de pacotes instalados) desnecessárias.

Sintaxe:

```
sudo apt autoremove
```

2.3.1.36.2. Gestor de pacotes dpkg

O gestor de pacote de baixo-nivel dpkg permite-nos instalar 'softwares' a partir de executáveis com a extensão .deb. Para usarmos este gestor, devemos usar o comando `-dpkg` em conjunto com certas opções conforme a necessidade, abaixo estão algumas opções básicas que podem ser usadas no dia a dia com o dpkg:

dpkg -i ou dpkg --install: o comando `-dpkg` com a opção **--install** ou **-i**, permite a instalação de pacotes .deb.

Sintaxe com opção não abreviada:

```
sudo dpkg --install nome_arquivo_executável.deb
```

Sintaxe com opção abreviada:

```
sudo dpkg -i nome_arquivo_executável.deb
```

dpkg -r ou dpkg --remove: o comando `-dpkg` com a opção **--remove** ou **-r** permite remover um determinado pacote, porém, o uso deste comando não garante a remoção completa do pacote do sistema. Deste modo, após a execução deste comando continuará armazenado no sistema informações associados aos arquivos de configuração, os scripts do responsável do pacote, arquivos de log (logs do sistema) e outros dados do utilizador manipulados pelo

pacote. Para remover completamente o pacote devemos usar a opção do comando **--purge** ou **-p**.

Sintaxe com a opção não abreviada:

```
sudo dpkg --remove nome_do_pacote.deb
```

Sintaxe com a opção abreviada:

```
sudo dpkg -r nome_do_pacote.deb
```

dpkg -l ou dpkg --list: o comando `-dpkg` com a opção **-l** ou **--list** é um comando de consulta da base de dados dpkg do sistema, este retornará uma lista de pacotes instalados com informações do pacote acerca da sua versão, a sua arquitetura e a descrição do pacote.

Sintaxe com a opção não abreviada:

```
sudo dpkg --list
```

Sintaxe com a opção abreviada:

```
sudo dpkg -l
```

dpkg -p ou dpkg --purge: o comando `-dpkg` com a opção **-p** ou **--purge** permite remover completamente o pacote instalado do sistema, incluindo as informações associadas aos arquivos de configuração, os scripts do responsável do pacote, arquivos de log (logs do sistema) e outros dados do utilizador manipulados pelo pacote.

Sintaxe com a opção não abreviada:

```
sudo dpkg --purge
```

Sintaxe com a opção abreviada:

```
sudo dpkg -p
```

2.3.1.37. Comandos de rede

Os comandos de rede são comandos que nos permite fazer a gestão dos serviços de rede, obter informações associados aos serviços de rede, entre outras ações. Alguns comandos de rede poderão não vir pré-instalados em determinadas distribuições GNU/Linux, havendo a necessidade de instalá-las para ter proveito delas.

2.3.1.37.1. Comando ifconfig

O comando `-ifconfig` é um comando de rede utilizado para verificar os detalhes da rede onde está conectada como, por exemplo: IP da máquina, máscara de rede etc.

Sintaxe:

```
ifconfig
```

2.3.1.37.2. Comando ifdown

O comando de rede `-ifdown` permite desativar a placa de rede.

Sintaxe:

```
ifdown "nome da placa de rede"
```

2.3.1.37.3. Comando ifup

O comando de rede `-ifup` permite habilitar a placa de rede.

Sintaxe:

```
ifup "nome da placa de rede"
```

2.3.1.37.4. Comando ping

O comando de rede `-ping` serve para testar a comunicação/conexão com outro host ou IP.

Sintaxe:

```
ping <endereço_ip_ou_domínio>
```

2.3.1.37.5. Comando netstat

O comando de rede `-netstat` permite visualizar as rotas.

Sintaxe:

```
netstat
```

2.3.1.37.6. Comando whois

O comando de rede `-whois` retorna informações de um domínio.

```
whois
```

2.3.1.37.7. Comando nslookup

O comando de rede `-nslookup` permite resolver domínios e trazer informações sobre o endereço de domínio.

Sintaxe:

```
nslookup endereço_de_domínio
```

2.3.2. Referência global

Os caracteres de referência global são recursos para adivinhar um nome de arquivo ou, especificar facilmente um ou mais grupos de nome de arquivos, ou diretórios do sistema de uma só vez.

Existem quatro (4) caracteres de referência global que podem ser utilizados individualmente ou em conjunto, nomeadamente:

'*' ⇒ este carácter de referência global usa-se para referir a vários caracteres. Ou seja, qualquer carácter dentro de um determinado parâmetro.

Exemplo:

```
ls /etc/f*
```

'**?**' ⇒ este carácter de referência global usa-se para referir a uma determinada letra naquela posição. Ou seja, substitui apenas uma letra.

Exemplo:

```
ls ????.pdf
```

‘[]’ ⇒ este carácter de referência global usa-se para referir aos caracteres ou símbolos que estiverem nos colchetes que podem ser combinados.

Exemplo:

```
ls [cC]*
```

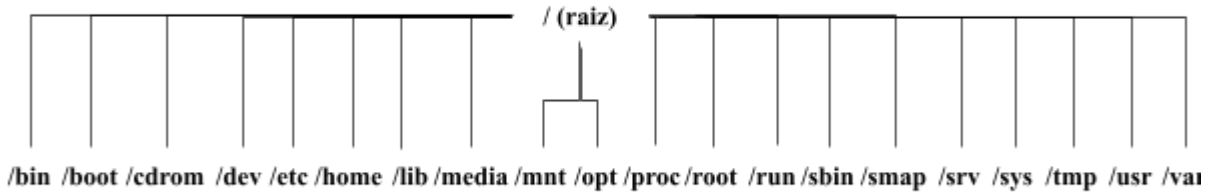
‘{}’⇒ este carácter de referência global usa-se para a procura de arquivos que terminam com uma sequência de carateres. Ideal para extensões.

Exemplo:

```
ls *.{pdf,docx}
```

2.4. Diretórios Linux

A estrutura de sistema de ficheiros do GNU/Linux é definida por um padrão denominado por **FHS, — Filesystem Hierarchy Standard (Padrão Hierarquia do Sistema de Arquivos)**, este permite que as diversas distribuições tenham uma estrutura de diretórios padrão para o sistema de arquivos permitindo uma maior consistência entre os sistemas e distribuições. Os diretórios do GNU/Linux tem uma estrutura específica, onde, temos o diretório raiz representada por ‘/’, nele contêm os diretórios do sistema e para o armazenamento dos arquivos do utilizador do computador, sugere-se que se armazene no diretório ‘/home’. Abaixo está um pequeno esquema da estrutura de diretórios.



Diretório /bin (binários essenciais) : neste diretório ficam armazenados arquivos binários, nele encontramos vários executáveis de diversos programas do sistema operativo.

Diretório /boot (ficheiros de arranque do sistema) : neste diretório encontramos diversos arquivos necessários para que o arranque do sistema operativo seja possível. Esta pasta é muito importante e não deve ser modificada.

Diretório /cdrom (ficheiros de cd) : este diretório, é um diretório legado. Apenas tem uso quando temos um pc com drivers cd-rom, é a pasta onde aparece os dados do cd-rom.

Diretório /dev (arquivos de dispositivos) : este diretório é para os dispositivos, cada arquivo dentro deste repositório corresponde a um dispositivo/hardware.

Diretório /etc (arquivos de configuração) : este diretório tem como principal objetivo armazenar os arquivos de configuração do sistema de forma "system wide", isto é, para todos os utilizadores do sistema.

Diretório /lib (arquivos de bibliotecas e módulos do kernel): este diretório contém bibliotecas de 'software', do sistema operativo ou qualquer programa instalado. É também neste diretório que contém os módulos do kernel do sistema operativo.

Diretório /media (unidades removíveis do sistema) : neste diretório serão montados automaticamente as unidades removíveis do sistema, exemplo: pen drivers, discos rígidos externos, etc.

Diretório /mnt (pontos de montagem temporários) : este diretório pode ser considerado "diretório irmão" do diretório /media foi projetado para ser o ponto de montagem no disco efetuado pelo próprio utilizador manualmente.

Diretório /opt (arquivos de 'softwares' instalados): este diretório refere-se a palavra opcional, é nela que encontramos 'softwares' instalados por fabricantes que enviam computadores instalados com sistema operativo com Linux ou por 'softwares' proprietários que querem organizar as suas informações principais num único diretório.

Diretório /proc (arquivos de sistema) : neste diretório encontramos arquivos que contém informações sobre o sistema e processos do mesmo. É um diretório virtual.

Diretório /root (arquivo utilizador root) : este diretório é para o utilizador root idêntico ao diretório /home.

Diretório /run (arquivos diversos) : este diretório, é um diretório virtual, carregado na memória do computador e apagado toda a vez que o computador é desligado. Armazena informação desde a última inicialização (boot), utilizadores com sessão iniciada, etc.

Diretório /sbin (binários pelo usuário root) : este diretório, é um diretório que armazena binários como o diretório /bin, com a diferença que o diretório /sbin tem programas que apenas quem tem permissão de executar é o superadministrador root do sistema.

Diretório /smap (arquivos de pacotes smap) : neste diretório contém arquivos para os pacotes smap que a canonical desenvolve.

Diretório /srv (serviços) : este diretório geralmente serve para serviços de servidor.
Diretório /sys : este diretório permite interagir diretamente com kernel do sistema operativo GNU/Linux.
Diretório /tmp (arquivos temporários) : este diretório armazena arquivos temporários.
Diretório /usr (Programas e aplicações de utilizadores) : este é um diretório de programas e bibliotecas úteis para o utilizador do sistema e não vitais para o funcionamento do sistema. Os programas instalados via código-fonte geralmente vão parar neste diretório.
Diretório /var (arquivos e dados variáveis) : este é um diretório de variáveis, que armazena arquivos que se esperam que aumentem de tamanho ao longo do tempo.

2.5. Instalação de servidor Web Apache2

Para a instalação do servidor web Apache2, caso ainda não esteja instalado, devemos executar no terminal o comando abaixo a este parágrafo. Quando utilizamos o parâmetro **-y** irá nos permitir que a instalação seja feita sem interrupções de questões durante a instalação do serviço na máquina.

```
apt-get install apache2 ou apt-get -y install apache2
```

Após executar o comando acima no terminal devemos verificar se o serviço já foi inicializado na máquina, para isso devemos executar o comando abaixo para ver o status atual do serviço.

```
service apache2 status
```

Existem alguns comandos associados ao controlo de serviço apache2 que devesse saber nomeadamente para parar, iniciar, consultar e reiniciar o serviço.
Comando para parar o serviço:

```
service apache2 stop
```

Comando para iniciar o serviço:

```
service apache2 start
```

Comando para reiniciar o serviço:

```
service apache2 restart
```

Comando para saber o estado do serviço;

```
service apache2 status
```

Após a instalação do serviço de servidor web na máquina será criada um diretório /www localizado no diretório /var (/var/www).

Considerações finais

Dando término a esta apostila, venho agradecer a quem teve a vontade de o ler e fazer um pedido aos leitores que em caso de quererem partilhar este conteúdo para fins académicos ou outros fins, que não se esqueçam de mencionar os direitos autorais e fonte onde adquiriram este conteúdo.