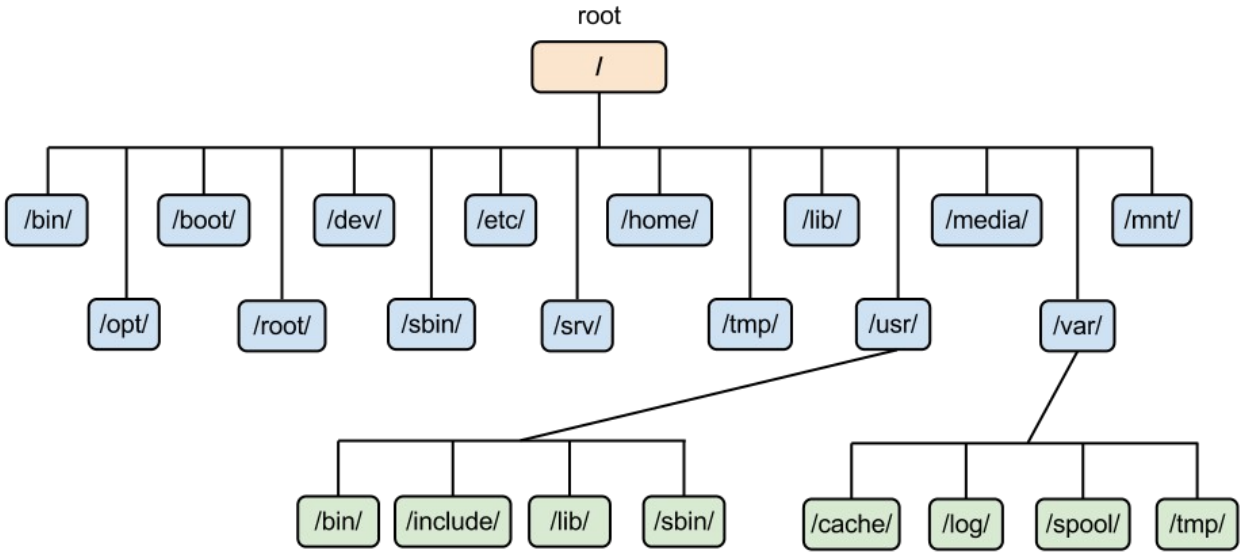
Manipulação de Arquivos e Pastas

1. Hierarquia de arquivos e pastas no Linux

A hierarquia de pastas e arquivos no Linux funciona de um modo diferente do que se é conhecido através do Windows.

Não existe o conceito de C:\ , ou D:\, etc. Existe somente uma hierarquia de arquivos, e qualquer dispositivo como pendrives e discos são mapeados dentro dessa hierarquia.

Isso fica mais claro na figura abaixo. Verifique que tudo começa a partir da raiz, / ou também root (nesse caso, o root não é o usuário administrador root, mas sim a raiz do sistema de arquivos). A raiz é o nível mais alto de hierarquia do sistema de arquivos. Por exemplo, considerando seu windows, é como se você estivesse abrindo a pasta do "Meu Computador" e a partir de lá você acessasse C:\, D:\ ou seu DVD.



O "/" (barra) é considerado um diretório/pasta e dentro dele existem várias outras pastas. O nome dessas pastas pode seguir um modelo diferente, dependendo de sua distribuição Linux. Algumas distribuições tentam até mapear o mesmo modelo já conhecido no Windows. Mas em geral, a maioria das distribuições apresentam a hierarquia acima ilustrada.

Como pode ser visto, cada pasta começa com o "/", isso indica que tal diretório está localizado/armazenado no diretório raiz ("/"). O '/' (barra) serve também para separar os nomes dos diretórios, assim como ocorre no Windows, onde o separador é o '\'.

Cada uma dessas pastas acima tem um objetivo diferente. Algumas servem para guardar arquivos de configuração, outras para programas, outras são pastas que não tem conteúdo lógico armazenado em disco, sendo montadas diretamente em memória. Abaixo vamos descrever todas essas pastas.

Diretório	Descrição
/bin	Nesta pasta você encontrará os executáveis/programas básicos do linux, como ‘cp’, ‘mv’, ‘ls’, ‘find’, etc. É uma pasta onde todos os usuários tem acesso de leitura e execução, de modo que possam acessar esses programas a partir do terminal.
/boot	Armazena informações de boot (quais sistemas operacionais você possui, em qual partição ou disco eles estão, etc), programas de teste de memória e também os Kernels de seu Linux.
/dev	É uma pasta que não armazena arquivos em si, mas serve para mapear todos os dispositivos que você possui em seu computador, como placas, discos, pendrives, cdrom, memória, etc. Cada dispositivo possui uma representação em formato de arquivo dentro deste diretório.
/etc	Utilizado para armazenar arquivos de configuração. Esses arquivos de configuração podem ser para serviços de rede, configuração de IP, montagem automática de discos, configurações para seu ambiente gráfico, configurações de drivers, scripts de inicialização do Linux, dentre muitos outros.
/home	Dentro deste diretório cada usuário pode possuir um diretório com seu nome. Por exemplo, o usuário “mariano” tem uma pasta ‘/home/mariano’ onde estão todos seus arquivos pessoais. De modo semelhante ao Windows, este diretório corresponde ao ‘C:\Users’ ou ‘C:\Documents and Settings’, armazenando pastas de todos os usuários e seus documentos, downloads, músicas, etc.
/lib	Armazena bibliotecas e módulos importantes ao Linux.
/media	Neste diretório, algumas distribuições, como o Ubuntu, fazem a montagem automática de pendrives ou discos conectados ao seu computador. O sistema cria um diretório dentro de /media para cada novo dispositivo conectado.
/mnt	É um diretório também utilizado como ponto de montagem genérico, mas a montagem de dispositivo é realizada manualmente. Exemplo: você deseja montar uma partição de disco em algum lugar, ao invés de criar um diretório para isso, você pode utilizar o /mnt temporariamente.
/opt	Armazena arquivos de softwares de terceiros. Normalmente softwares de terceiros que possuem muitos arquivos são armazenados nesta pasta.
/proc	Armazena dados e estatísticas do escalonador e de todos os processos em execução no sistema operacional.
/root	É o diretório do usuário ‘root’ (o administrador geral do sistema). Assim como cada usuário possui um diretório em ‘/home’, o usuário ‘root’ possui a pasta ‘/root’ para armazenar seus arquivos.
/sbin	Armazena aplicativos do superusuário. São ferramentas administrativas que devem ser acessadas apenas pelo usuário root ou usuários com privilégios de administrador.
/tmp	Consiste em uma pasta com leitura e escrita liberada a todos os usuários. Esta pasta é utilizada por softwares como diretório temporário para armazenar temporariamente arquivos que estão sendo utilizados, compiados, compactados, etc. Arquivos importantes não devem ser salvos nesta pasta.
/usr	Dentro deste diretório são armazenados arquivos multi-usuários e também aplicações e bibliotecas de terceiros.
/var	Armazena logs e arquivos gerados a partir da execução de serviços e outros softwares.

2. Caminhos absolutos e relativos

Todo diretório no Linux possui 2 diretórios especiais o ‘.’ (ponto) e o ‘..’ (ponto ponto). Você pode verificar isso utilizando o comando ‘ls -la’ a partir do terminal. Esses dois diretórios especiais estão relacionados ao diretório corrente e ao diretório pai de seu diretório atual.

Por exemplo, se em seu terminal aparece isso: “usuario@host:/usr/bin”, significa que você está logado com usuário ‘usuario’ na máquinha ‘host’ e seu diretório corrente é o ‘/usr/bin’.

Se você executar o comando ‘ls -la’ neste diretório, você verá vários arquivos, dentre eles, lá no começo da lista estão os diretórios especiais ‘.’ e ‘..’. O diretório ‘.’ corresponde ao seu diretório corrente, ou seja, ‘/usr/bin’. O diretório ‘..’ corresponde ao diretório acima, ou seja, ‘/usr’. Esse dois diretórios especiais são utilizados quando é desejado especificar um caminho relativo. O caminho relativo é dito como tal por especificar o caminho a partir de seu path atual, ou seja, ‘/usr/bin’. Se você deseja acessar o ‘/usr’, seu caminho relativo será ‘..’ ou ‘../’ (com o barra no final). Se você deseja acessar o diretório ‘/’ a partir do ‘/usr/bin’, basta utilizar ‘../..’ ou ‘../../’.

Já o caminho absoluto não leva em consideração qual o seu diretório atual, ele sempre inicia a partir do ‘/’. O ‘/usr/bin’ é um exemplo de um caminho absoluto, pois inicia com o símbolo da raiz ‘/’.

3. Comando ‘ls’ e variações

O comando ‘ls’ é utilizado para listar todos os arquivos ou diretórios de um determinado caminho. Caso este caminho seja omitido, comando ‘ls’ deve listar os arquivos e diretórios de seu diretório atual.

Veja abaixo alguns exemplos:

ls	Modo mais simples, somente lista arquivos e diretórios de seu diretório atual.
ls *.txt	Lista todos os arquivos com extensão ‘.txt’ em seu diretório atual.
ls /tmp	Lista os arquivos do diretório /tmp
ls /tmp/*.txt	Lista todos os arquivos com extensão ‘.txt’ do caminho ‘/tmp’.
ls -l	Saída no formato de lista (uma única coluna), com nome do arquivo, data de modificação e atributos de acesso.
ls -la	Saída no formato de lista e também mostrando arquivos ocultos, cujos nomes são iniciados com o ‘.’
ls -lt	Mesmo que ‘ls -l’, mas ordenando por data/hora de última modificação (mais novo primeiro)
ls -ltr	Mesmo que o anterior, fazendo com que a order seja reversa, ou seja, mais novo por último.
ls .. -l	Lista os arquivos do diretório superior ao atual no formato de lista (coluna única com data/hora, tamanho, nome e atributos)
ls -lS	Ordena por tamanho (maior primeiro)

ls -lSr	Mesmo que o anterior, de ordem reversa, ou seja, menor primeiro.

4. Criação de pastas e navegação

A navegação entre pastas é realizada no terminal por meio do comando ‘cd’.

Este comando recebe como argumento um caminho que pode ser absoluto ou relativo. A partir da execução, seu diretório corrente deve ser trocado para o diretório apontado pelo caminho passado como parâmetro para o comando.

Veja abaixo alguns exemplos de utilização desse comando.

cd /tmp	Muda diretório corrente para o ‘/tmp’.
cd ..	Muda o diretório corrente para o diretório acima. Se você estiver no diretório ‘/usr/lib’ e digitar ‘cd ..’, seu novo diretório atual será ‘/usr/’.
cd /usr/lib	Não importa seu diretório corrente, seu novo diretório corrente será ‘/usr/bin’, pois você está passando um caminho absoluto como parâmetro para o comando ‘cd’.
cd ../..	Se você estiver dentro de /usr/lib e digitar este comando, seu diretório corrente será o ‘/’.
cd -	Vai para o último diretório antes da execução do último comando ‘cd’.
cd ~ ou cd	O ‘~’ é um alias para seu diretório de usuario. Se o seu usuário é ‘lucas’ e você digitar ‘cd ~’ ou apenas ‘cd’, não importa qual seu diretório corrente, seu novo diretório será ‘/home/lucas’. Por isso que no terminal aparece ‘lucas@host:~’, o ‘~’ significa que você está em seu diretório *home*.

Para criar um novo diretório você pode utilizar o comando ‘mkdir’. Este comando recebe um parâmetro que é o nome do novo diretório a ser criado.

Exemplo: ‘mkdir novodiretorio’ - este comando criará dentro de seu diretório corrente o diretório ‘novodiretorio’.

Você pode também especificar um caminho onde seu diretório será criado. Por exemplo: ‘mkdir /tmp/novodiretório’. O comando cria, dentro do diretório ‘/tmp’, o diretório ‘novodiretorio’.

5. Renomeando, movendo e copiando arquivos

a) Renomeando arquivos

Para renomear arquivos, você pode utilizar o comando ‘mv’. Este comando, assim como explicado nos próximos parágrafos, também é utilizado para mover arquivos ou diretórios em seu sistema de arquivos (o mesmo comando é utilizado, porque internamente, renomear ou mover é realizado do mesmo modo).

A sintaxe do comando ‘mv’ para renomear recebe dois parâmetros, o primeiro que compreende o arquivo ou diretório a ser renomeado, e o segundo que denota o novo nome deste arquivo. Alguns exemplos são demonstrados na tabela a seguir.

mv file1 file2	Renomeia o arquivo ‘file1’ para ‘file2’.
mv /tmp/file1 /tmp/file2	Renomeia o arquivo ‘file1’, que está dentro do diretório ‘/tmp’, para ‘file2’
mv dir1 dir2	A mesma sintaxe é aplicada a diretórios. O primeiro parâmetro é o diretório que se deseja renomear e o segundo parâmetro é o novo nome.

b) Movendo arquivos

A sintaxe do comando ‘mv’ é a mesma para mover arquivos. O primeiro parâmetro deve ser o nome do arquivo a ser movido, enquanto o segundo consiste no novo local / diretório onde esse arquivo será armazenado. Veja alguns exemplos:

mv arquivo /tmp	Movendo ‘arquivo’ da pasta corrente para a pasta /tmp, utilizando path absoluto.
mv arquivo ..	Move ‘arquivo’ para um diretório acima de seu diretório corrente.
mv arquivo /tmp/arquivo1	Move ‘arquivo’ para a pasta /tmp, renomeando-o para ‘arquivo1’
mv ./arquivo /tmp	Exatamente igual ao primeiro exemplo. Move ‘arquivo’ do diretório corrente para a pasta ‘/tmp’
mv ../arquivo .	Move ‘arquivo’ que está no diretório acima para o diretório corrente.
mv /tmp/arquivo ./	Move ‘arquivo’ dentro de ‘/tmp’ para o diretório corrente.

c) Copiando arquivos

Para copiar arquivos ou pastas, o comando ‘cp’ deve ser utilizado.

O comando ‘cp’ aceita várias opções, mas dois parâmetros são essenciais: O que deve ser copiado e para Onde deve ser copiado.

Abaixo estão descritos alguns exemplos de utilização desse comando.

cp arquivo1 arquivo2	Faz uma cópia do ‘arquivo1’, localizado na pasta corrente, para ‘arquivo2’, também na pasta corrente.
cp arquivo1 /tmp	Copia ‘arquivo1’ para a pasta ‘/tmp’. Como o nome do arquivo final foi omitido, o mesmo nome será utilizado no arquivo final em ‘/tmp’.

cp arquivo1 /tmp/arquivo2	Mesmo que o anterior, mas agora especificamos que o arquivo em ‘/tmp’ será chamado ‘arquivo2’.
cp ../arquivo1 .	Copia ‘arquivo1’ localizado na pasta acima da pasta corrente para a própria pasta corrente (‘.’ diretório corrente).
cp arquivo1 ../arquivo2	Copia o arquivo da pasta corrente chamado ‘arquivo1’ para a pasta acima, chamando-o de ‘arquivo2’.
cp -r pasta1 pasta2	Utilizamos o parâmetro ‘-r’, fazendo com que o comando ‘cp’ realize uma cópia recursiva de um diretório chamado ‘pasta1’ para um novo diretório ‘pasta2’ que conterá todos seus arquivos. É importante lembrar que este parâmetro deve ser utilizado sempre que uma cópia de diretórios deve ser realizada.
cp -r /tmp/pasta1 .	Copia a pasta ‘/tmp’ para o diretório corrente.
cp -r ../pasta2 ./pasta1	Copia a pasta localizada um diretório acima chamada ‘pasta2’ para o diretório corrente, chamando-a de ‘pasta1’

6. Busca recursiva de arquivos e textos em arquivos

a) Busca de arquivos

O principal comando para localizar um determinado arquivo é o comando ‘find’. Este comando tem como parâmetros: o local a partir do qual o arquivo será procurado e o nome do arquivo.

Vejamos alguns exemplos.

Queremos procurar pelo arquivo: notas.txt no diretório corrente. A sintaxe do nosso comando será a seguinte:

```
find . -name notas.txt
```

Caso você queira procurar também por ‘NOTAS.TXT’, ‘Notas.txt’ ou ‘notas.txt’, independente se as letras são maiúsculas ou minúsculas, você deve utilzar o parâmetro ‘-iname’ ao invés de ‘-name’. Sendo assim o novo comando ficará assim:

```
find . -iname notas.txt
```

Caso você queira encontrar qualquer arquivo cujo nome seja iniciado por ‘notas’, independente do restante do nome, você poderá utilizar o seguinte comando:

```
find . -name ‘notas*’
```

O ‘*’ é utilizado para indicar que naquela parte, tanto faz os caracteres que o compõem, o que importa é o início.

Se você deseja fazer uma busca no diretório ‘/tmp’, por exemplo, o comando ficará assim:

```
find /tmp -name ‘notas.*’
```

Neste caso estamos procurando por um arquivo chamado ‘notas’, não importando sua extensão.

b) Busca de textos em arquivos

O comando utilizado para buscar texto em arquivos é o ‘grep’. Ele possui uma grande quantidade de parâmetros, podendo também ser utilizado com expressões regulares. Abaixo vemos os principais exemplos de utilização deste comando.

grep ‘qualquer’ arquivo.txt	Procura pelo texto ‘qualquer’ dentro do arquivo ‘arquivo.txt’.
grep -i ‘qualquer’ file.txt	Procura pelo texto ‘qualquer’, com letras maiúsculas ou minúsculas dentro do arquivo ‘file.txt’.
grep ‘texto’ *.txt	Procura pelo texto ‘texto’ em todos os arquivos da pasta corrente que possuem extensão .txt.
grep ‘texto’ -r .	Procura pelo texto ‘texto’ recursivamente em todos os arquivos dentro da pasta corrente, ou dentro de outros diretórios na pasta corrente (determinado pelo ‘.’)
grep ‘teste’ -rn .	Procura pelo texto ‘teste’ recursivamente em todos os arquivos dentro do diretório corrente, ou dentro de outros diretórios na pasta corrente, evidenciando qual a linha de cada arquivo onde uma ocorrência de ‘teste’ foi encontrada.