

alpha

<ed/tech>

LINUX

Aula 02

<Módulo 07/>

Introdução

Abordaremos elementos essenciais relacionados ao gerenciamento eficiente de arquivos e dados. No decorrer deste material, aprenderemos sobre: **Advanced Package Tool (APT)** para instalação e atualização de pacotes; estratégias de compactação e descompactação para otimização de armazenamento; técnicas avançadas de busca de arquivos; e piping, que permite o encadeamento de comandos.

Comando sudo

O comando **sudo** é utilizado para executar comandos com privilégios elevados, geralmente associados à administração do sistema. A sigla "sudo" significa "**superuser do**" ou "**substitute user do**", indicando que você está temporariamente assumindo os privilégios de superusuário para realizar uma tarefa específica.

O superusuário, muitas vezes chamado de "root", possui permissões elevadas que permitem realizar ações críticas no sistema, como instalação e remoção de software, modificação de configurações essenciais e manipulação de arquivos do sistema. No entanto, para evitar a execução inadvertida de comandos perigosos que podem impactar o sistema, o acesso ao superusuário é restrito e requer autenticação.

Ferramenta apt

Distribuições Linux baseadas em Debian e suas variantes, como o Ubuntu utilizam o **apt** como uma ferramenta de linha de comando para o gerenciamento de pacotes. Ele facilita a instalação, remoção, atualização e gerenciamento de pacotes de software no sistema. A palavra "apt" é uma abreviação de **Advanced Package Tool** (ferramenta avançada de pacotes, em português).

Você talvez já tenha encontrado os comandos **apt** ou **apt-get**, os quais, embora parecidos, não são idênticos. O apt, como ferramenta de linha de comando, apresenta uma interface de usuário mais intuitiva para usuários finais, com comandos intuitivos, comportamentos resultantes e recursos de segurança. Por outro lado, o apt-get é uma interface de baixo nível que se comunica de maneira mais direta com os processos essenciais do sistema Linux. O apt é considerado um gerenciador de pacotes mais amigável em comparação com o apt-get.

As tarefas mais comuns no apt e no apt-get são instalar, atualizar e remover pacotes de software. Os pacotes de software podem ser aplicações, bibliotecas, utilitários e ferramentas. A sintaxe é muito parecida, por exemplo,

```
apt install <nome-do-pacote>  
apt-get install <nome do pacote>
```

A distinção entre apt e apt-get vai além do fato de que o apt é uma versão mais recente do apt-get. O comando apt foi desenvolvido como uma alternativa mais acessível ao apt-get, unificando a funcionalidade de diversas ferramentas de gerenciamento de pacotes para proporcionar maior conveniência ao usuário.

Comparação apt vs apt-get

As principais diferenças são:

- **Datas de Lançamento:** Em 1998, o apt-get foi introduzido com a distribuição Debian 2.0 (Hamm), enquanto o apt foi lançado em 2014 com a distribuição Debian 8 (Jessie). Posteriormente a 2014, o apt assumiu o lugar do apt-get como a ferramenta padrão de gerenciamento de pacotes para todas as distribuições Linux baseadas em Debian.
- **Recursos de Pesquisa:** Utilize o comando `apt search <nome-do-pacote>` para pesquisar um pacote pelo nome. Esse comando oferece uma descrição detalhada de todos os pacotes que contêm o nome do pacote pesquisado. Essa funcionalidade não estava disponível no apt-get, sendo necessário utilizar o comando `apt-cache <nome-do-pacote>` para realizar essa operação.
- **Resolução de Dependências:** Cada pacote de software geralmente possui uma lista de dependências, como bibliotecas ou ferramentas necessárias para seu correto funcionamento. A instalação dessas dependências é crucial antes de poder instalar o próprio pacote. Ambos o apt e o apt-get lidam com a resolução de dependências, porém, o apt demonstra superioridade nesse aspecto. Ele gerencia cadeias de dependências complexas, instalando pacotes na ordem adequada e recomendando pacotes sugeridos para instalação.
- **Versões do Pacote no Sistema de Arquivos:** Por padrão, o comando `apt upgrade` remove versões antigas de pacotes instalados ou atualizáveis no sistema que não são mais necessárias durante a atualização. Em contrapartida, o comando `apt-get upgrade` não possui essa eficiência, tornando o `apt upgrade` mais eficaz para liberar espaço de memória no sistema.
- **Saídas Impressas no Terminal:** Ambos o apt-get e o apt imprimem informações de status no terminal, detalhando as ações do sistema após o comando de entrada. O comando `apt` fornece informações um pouco mais abrangentes para o usuário, incluindo uma barra de progresso em cada tarefa.

Sintaxe básica

A sintaxe básica para usar o apt é:

apt comando
apt [opções/flags] comando [pacotes]

Opção	Descrição
-h, --help	Imprime um guia de ajuda
-d, --download-only	Baixa um pacote, mas não continua com a instalação
--no-download	Não baixa nenhum pacote e usa os já baixados
--assume-no	Responde "não" a todas as solicitações
-y	Responde "sim" aos prompts sem interromper o processo
-f, --fix-broken	Tenta consertar dependências quebradas
-s, --simulate	Não altera o sistema, apenas exibe qual será a saída

Comandos

A ferramenta apt funciona com diversos comandos. Apresentamos os principais a seguir:

Comando	Descrição
update	Obtém informações sobre as versões mais recentes dos pacotes disponíveis, mas não instala nenhuma atualização.
upgrade	Baixa as versões atualizadas dos pacotes e atualiza os pacotes instalados para a nova versão.
full-upgrade	Atualiza os pacotes atualmente instalados e remove pacotes desnecessários para a atualização completa do sistema.
install	Instala um pacote especificado do repositório.
remove	Exclui um pacote, mas deixa os arquivos de configuração.
purge	Exclui um pacote e quaisquer arquivos de configuração.
autoremove	Remove dependências que não são mais necessárias.
list	Lista todos os pacotes disponíveis ou pacotes de acordo com critérios específicos.
search	Pesquisa pacotes cujo nome ou descrição contém o termo de pesquisa.
show	Imprime detalhes sobre um pacote específico.
edit-sources	Permite que os usuários editem fontes de pacotes em um editor de texto.

É fundamental ressaltar que a utilização dos comandos apt exige a inclusão do sudo para conclusão, especialmente quando são necessárias permissões de root para acessar, modificar ou executar arquivos. A situação é ilustrada na imagem a seguir, na qual o comando apt update foi executado sem a adição do sudo (superuser do), resultando em uma mensagem de "permissão negada".

```
letonio.silva@BRRIOLN043879:~ $ apt update
Lendo listas de pacotes... Pronto
E: Não foi possível abrir arquivo de trava /var/lib/apt/lists/lock - open (13: P
ermission denied)
E: Impossível criar acesso exclusivo ao directório /var/lib/apt/lists/
W: Problema ao remover o link do ficheiro /var/cache/apt/pkgcache.bin - RemoveCa
ches (13: Permission denied)
W: Problema ao remover o link do ficheiro /var/cache/apt/srcpkgcache.bin - Remov
eCaches (13: Permission denied)
letonio.silva@BRRIOLN043879:~ $
```

Algumas boas práticas:

- **Atualizar o sistema regularmente** - Executar sudo apt update seguido de sudo apt upgrade (ou sudo apt full-upgrade) regularmente, geralmente semanalmente, é uma prática recomendada. Isso garante que você tenha as últimas atualizações de segurança e correções de bugs instaladas em seu sistema.
- **Utilizar autoremove** - O comando sudo apt autoremove remove automaticamente pacotes que foram instalados como dependências, mas que não são mais necessários. Isso ajuda a manter o sistema limpo, removendo pacotes que não são mais utilizados. Após instalar algo é interessante aplicar esse comando.
- **Utilizar clean para Limpar Cache** - O comando sudo apt clean remove os arquivos do cache dos pacotes baixados durante as atualizações. Isso ajuda a liberar espaço em disco.
- **Reiniciar o sistema após atualizações do Kernel** - Após atualizações do kernel, é uma boa prática reiniciar o sistema para garantir que as mudanças entrem em vigor.

Exemplo 01 - Vim

O objetivo desta seção é instalar o **Vim**, um poderoso editor de texto, conhecido por sua eficiência e riqueza de recursos. Ele oferece modos distintos, incluindo um modo de visualização, um modo de edição e um modo de comando, tornando-o inicialmente desafiador para os novos usuários, mas extremamente eficiente para aqueles que se tornam proficientes nele.

Infos

Para exibir detalhes sobre o pacote que pretendemos instalar (Vim), como dependências, descrição do conteúdo, download e tamanho da instalação, fontes, etc., use:

apt show vim

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ apt show vim
Package: vim
Version: 2:8.2.3995-1ubuntu2.15
Priority: optional
Section: editors
Origin: Ubuntu
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Debian Vim Maintainers <team+vim@tracker.debian.org>
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Installed-Size: 4.025 kB
Provides: editor
Depends: vim-common (= 2:8.2.3995-1ubuntu2.15), vim-runtime (= 2:8.2.3995-1ubuntu2.15), libacl1 (>= 2.2.23), libc6 (>= 2.34), libgpm2 (>= 1.20.7), libpython3.10 (>= 3.10.0), libselinux1 (>= 3.1-), libsodium23 (>= 1.0.14), libtinfo6 (>= 6)
Suggests: ctags, vim-doc, vim-scripts
Homepage: https://www.vim.org/
Task: cloud-image, ubuntu-wsl, server, ubuntu-server-raspi, lubuntu-desktop
Download-Size: 1.735 kB
APT-Sources: http://br.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages
```

Atualização

Antes de proceder com a instalação, é uma boa prática aplicar **apt update** e **apt upgrade** para determinar e atualizar pacotes desatualizados. Faça isso regularmente. Aplique os comandos a seguir:

sudo apt update
sudo apt upgrade

Na imagem a seguir, nota-se que 19 pacotes podem ser atualizados.

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ sudo apt update
Atingido:1 http://br.archive.ubuntu.com/ubuntu jammy InRelease
Obter:2 http://br.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Atingido:3 http://dl.google.com/linux/chrome/deb stable InRelease
Atingido:4 https://download.docker.com/linux/ubuntu jammy InRelease
Atingido:5 http://br.archive.ubuntu.com/ubuntu jammy-backports InRelease
Obter:6 http://br.archive.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Atingido:7 http://packages.microsoft.com/repos/code stable InRelease
Atingido:8 https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/jammy pgadmin4 InRelease
Atingido:9 https://ppa.launchpadcontent.net/deadsnakes/ppa/ubuntu jammy InRelease
Atingido:10 https://ppa.launchpadcontent.net/emoraes25/cid/ubuntu jammy InRelease
Baixados 229 kB em 1s (168 kB/s)
Lendo listas de pacotes... Pronto
Construindo árvore de dependências... Pronto
Lendo informação de estado... Pronto
19 packages can be upgraded. Run 'apt list --upgradable' to see them.
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Na imagem a seguir, nota-se que o usuário precisa confirmar se deseja fazer o upgrade. Alternativamente, ao aplicar o comando de upgrade, poderíamos passar a flag -y (**sudo apt -y upgrade**) para que o sistema aceite automaticamente as mudanças sem pedir confirmação.

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ sudo apt upgrade
Lendo listas de pacotes... Pronto
Construindo árvore de dependências... Pronto
Lendo informação de estado... Pronto
Calculando atualização... Pronto
0 seguinte pacote foi instalado automaticamente e já não é necessário:
 vim-runtime
Utilize 'sudo apt autoremove' para o remover.
Os pacotes a seguir serão atualizados:
 alsa-ucm-conf code containerd.io dnsmasq-base docker-ce docker-ce-cli docker-ce-rootless-extras
 docker-compose-plugin google-chrome-stable ldap-utils libldap-2.5-0 libldap-common linux-firmware
 python3-pil xserver-common xserver-xephyr xserver-xorg-core xserver-xorg-legacy xwayland
19 pacotes atualizados, 0 pacotes novos instalados, 0 a serem removidos e 0 não atualizados.
9 atualizações de segurança de LTS
É preciso baixar 562 MB/566 MB de arquivos.
Depois desta operação, 37,4 MB adicionais de espaço em disco serão usados.
Você quer continuar? [S/n] s
```

Após a atualização dos pacotes, é interessante remover dependências que não são mais necessárias, através desse comando:

sudo apt autoremove

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ sudo apt autoremove
Lendo listas de pacotes... Pronto
Construindo árvore de dependências... Pronto
Lendo informação de estado... Pronto
Os pacotes a seguir serão REMOVIDOS:
 vim-runtime
0 pacotes atualizados, 0 pacotes novos instalados, 1 a serem removidos e 0 não atualizados.
Depois desta operação, 33,6 MB de espaço em disco serão liberados.
Você quer continuar? [S/n] s
```

É possível executar os comandos update e upgrade juntos em uma única etapa e evitar a solicitação de confirmação do processo, use o a flag -y:

sudo apt update && sudo apt upgrade -y

Instalação

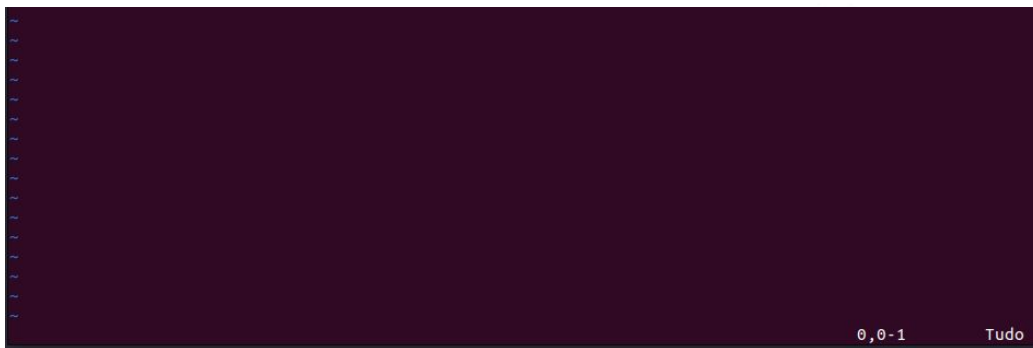
Para fazer a instalação do editor Vim, aplica-se o comando a seguir:
sudo apt install vim

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ sudo apt -y install vim
Lendo listas de pacotes... Pronto
Construindo árvore de dependências... Pronto
Lendo informação de estado... Pronto
The following additional packages will be installed:
  vim-runtime
Pacotes sugeridos:
  ctags vim-doc vim-scripts
```

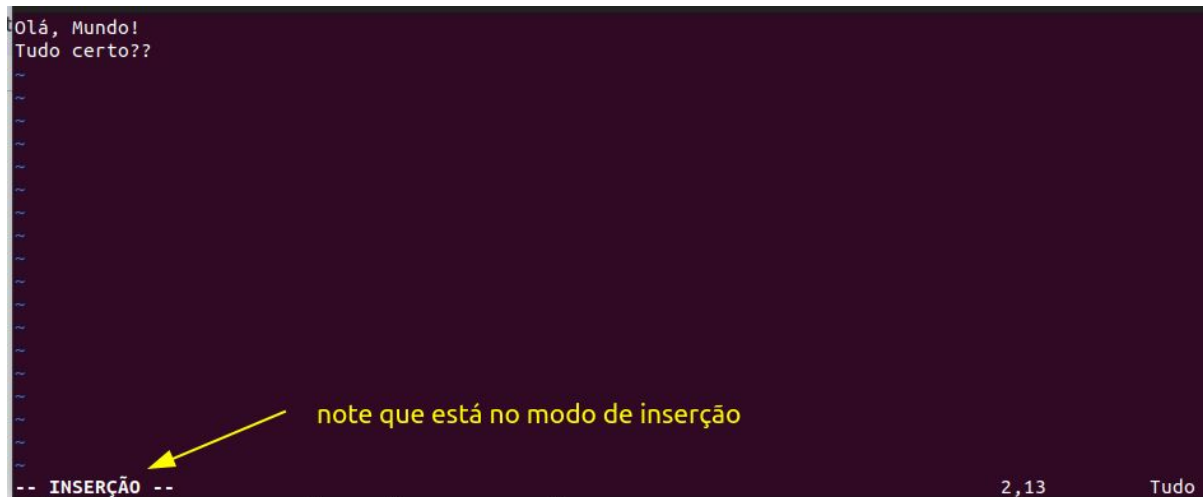
Uso

Vamos utilizar o Vim, iniciando um arquivo vazio chamado exemplo.txt, conforme esse comando:
vim exemplo.txt

Ao iniciar, o vim está no modo de comando (command mode).



Pressione a tecla "i" para entrar no modo de inserção. Em seguida, adicione algum texto.



Desinstalar

Suponhamos que, por alguma razão, você não esteja satisfeito com o Vim e deseje desinstalá-lo. Nesse caso, você pode optar por utilizar **remove** ou **purge**. Ao empregar **apt remove**, os arquivos de configuração do Vim são mantidos, preservando assim quaisquer personalizações que você tenha realizado. Isso engloba ajustes específicos, configurações personalizadas ou modificações feitas de acordo com suas preferências. Se, em algum momento, você optar por reinstalar o pacote, suas configurações personalizadas serão mantidas. Por outro lado, ao utilizar **apt purge**, tudo é removido, incluindo os arquivos de configuração. Essa abordagem garante uma desinstalação completa do Vim, sem retenção de quaisquer personalizações anteriores. Vamos aplicar o **apt remove**, conforme na imagem abaixo:

sudo apt remove vim

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ sudo apt remove vim
Lendo listas de pacotes... Pronto
Construindo árvore de dependências... Pronto
Lendo informação de estado... Pronto
O seguinte pacote foi instalado automaticamente e já não é necessário:
  vim-runtime
Utilize 'sudo apt autoremove' para o remover.
Os pacotes a seguir serão REMOVIDOS:
  vim
0 pacotes atualizados, 0 pacotes novos instalados, 1 a serem removidos e 0 não atualizados.
Depois desta operação, 4.025 kB de espaço em disco serão liberados.
Você quer continuar? [S/n] s
(Lendo banco de dados ... 241290 ficheiros e directórios actualmente instalados.)
A remover vim (2:8.2.3995-1ubuntu2.15) ...
update-alternatives: a usar /usr/bin/vim.tiny para disponibilizar /usr/bin/vi (vi) em modo auto
update-alternatives: a usar /usr/bin/vim.tiny para disponibilizar /usr/bin/view (view) em modo auto
update-alternatives: a usar /usr/bin/vim.tiny para disponibilizar /usr/bin/ex (ex) em modo auto
update-alternatives: a usar /usr/bin/vim.tiny para disponibilizar /usr/bin/rview (rview) em modo auto
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Novamente, aplicar autoremove é uma boa prática (usar a opção **-y** pula qualquer etapa de confirmação na remoção de dependências).

sudo apt -y autoremove

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ sudo apt -y autoremove
Lendo listas de pacotes... Pronto
Construindo árvore de dependências... Pronto
Lendo informação de estado... Pronto
Os pacotes a seguir serão REMOVIDOS:
  vim-runtime
0 pacotes atualizados, 0 pacotes novos instalados, 1 a serem removidos e 0 não atualizados.
Depois desta operação, 33,6 MB de espaço em disco serão liberados.
(Lendo banco de dados ... 241281 ficheiros e directórios actualmente instalados.)
A remover vim-runtime (2:8.2.3995-1ubuntu2.15) ...
A remover 'desvio de /usr/share/vim/vim82/doc/help.txt para /usr/share/vim/vim82/doc/help.txt.vim-tiny por vim-runtime'
A remover 'desvio de /usr/share/vim/vim82/doc/tags para /usr/share/vim/vim82/doc/tags.vim-tiny por vim-runtime'
A processar 'triggers' para man-db (2.10.2-1) ...
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Exemplo 02 - tree

Uma forma de exibir diretórios, subdiretórios e arquivos do ambiente atual de forma hierárquica e visualmente agradável é através da ferramenta **tree**. Basta aplicar o comando **tree** (caso não esteja instalado, use **sudo apt install tree**).

```
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $ tree
.
├── css
│   ├── hello.txt
│   ├── re-html
│   │   └── index.html
│   └── style.css
├── html
│   └── index.html
└── README.md

3 directories, 5 files
letonio.silva@BRRIOLN043879:~/Documentos/exemplo $
```

Exemplo 03 - Apache

Caso não esteja acessando a sua máquina virtual, faça o acesso agora. O objetivo é instalar o apache nela. Faça o acesso da maneira que achar mais confortável. Na imagens desta seção, optou-se por apenas abrir o aplicativo da VirtualBox e liguei a VM. Com a máquina virtual ligada, passe as credenciais do seu usuário cadastrado.

Em um dado momento, saber o ip da sua máquina virtual será importante. Utilize o seguinte comando para visualizar:

ip a

Arquivo	Máquina	Visualizar	Entrada	Dispositivos	Ajuda
lets@serverlets:~\$ ip a					
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000					
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00					
inet 127.0.0.1/8 scope host lo					
valid_lft forever preferred_lft forever					
inet6 ::1/128 scope host					
valid_lft forever preferred_lft forever					
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000					
link/ether 08:00:27:00:c5:86 brd ff:ff:ff:ff:ff:ff					
inet 192.168.0.102/24 metric 100 brd 192.168.0.255 scope global dynamic enp0s3					
valid_lft 85595sec preferred_lft 85595sec					
inet6 fe80::a00:27ff:fe00:c586/64 scope link					
valid_lft forever preferred_lft forever					
lets@serverlets:~\$ _					

Instalação

Para fazer a instalação do apache, aplique o seguinte comando:

sudo apt install apache2

```
lets@serverlets:~$ sudo apt install apache2
[sudo] password for lets: _
```

Status do serviço

Para verificar o status do apache, aplique esse comando:

sudo systemctl status apache2

```
lets@serverlets:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2024-02-10 22:29:09 UTC; 1min 29s ago
     Docs: https://httpd.apache.org/docs/2.4/
    Main PID: 2468 (apache2)
      Tasks: 55 (limit: 2221)
     Memory: 5.0M
        CPU: 23ms
    CGroup: /system.slice/apache2.service
            └─2468 /usr/sbin/apache2 -k start
              └─2470 /usr/sbin/apache2 -k start
                └─2471 /usr/sbin/apache2 -k start

Feb 10 22:29:09 serverlets systemd[1]: Starting The Apache HTTP Server...
Feb 10 22:29:09 serverlets apachectl[2467]: AH00558: apache2: Could not reliably determine the ser
Feb 10 22:29:09 serverlets systemd[1]: Started The Apache HTTP Server.
lines 1-16/16 (END)
```

Percebe-se na imagem acima que o serviço está ativo. Basta apertar Q (quit) para sair da tela e voltar ao terminal padrão. Caso não esteja, você pode aplicar esse comando:

sudo systemctl start apache2

Por padrão, o Ubuntu Server tem o Uncomplicated Firewall(ufw), que pode ou não estar ativo. Veja o status do firewall através desse comando:

sudo ufw status

Caso esteja ativo, será necessário habilitar o tráfego na porta 80. Caso esteja inativo não precisa fazer nada com relação a isso. O comando para habilitar na porta 80 é:

sudo ufw allow 80

Na imagem a seguir, mesmo inativo, o usuário optou por aplicar o comando apenas como demonstração. As regras relacionadas ao ipv4 e ipv6 foram atualizadas.

```
lets@serverlets:~$ sudo ufw status
Status: inactive
lets@serverlets:~$ sudo ufw allow 80
Rules updated
Rules updated (v6)
lets@serverlets:~$
```

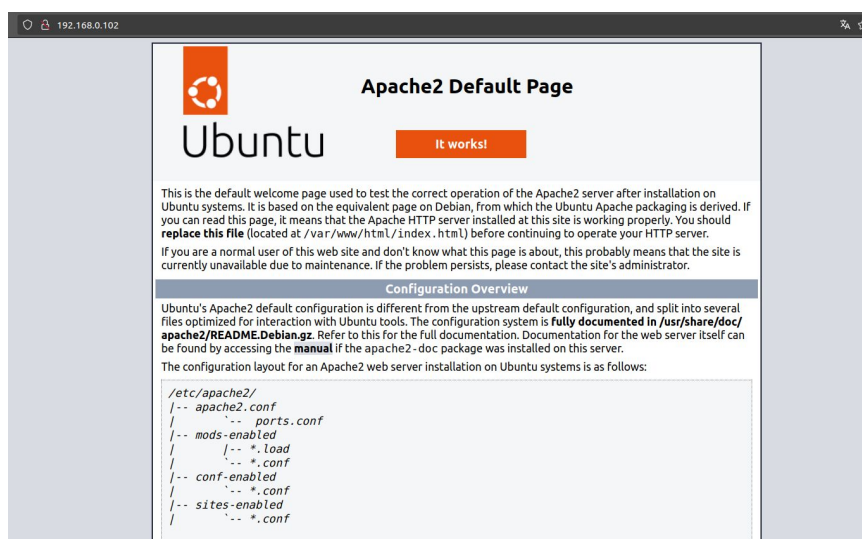

Página web

Quando o Apache é instalado e seu serviço está ativo, ele deixa uma página estática padrão sendo servida na porta 80. Consequentemente, sua máquina virtual passa a ser um servidor na sua rede local. Na sua máquina local e/ou pelo celular, abra um navegador web e insira o endereço seguindo esse padrão:

`http://<endereço_ip_da_maquina_virtual>`

Não é necessário adicionar a porta 80, pois o protocolo **http** define a 80 como a porta padrão, portanto, `http://<numero-ip>` e `http://<numero-ip>:80` são equivalentes.

Na imagem abaixo, observa-se a página estática que é servida pelo Apache.



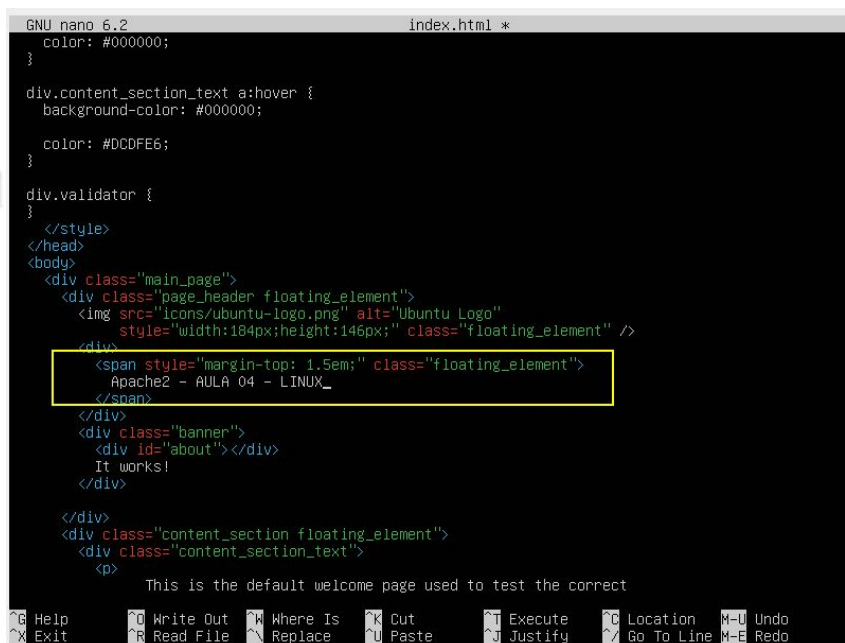
Você pode estar se perguntando: onde está o arquivo `index.html` dessa página que o Apache está servindo? Pois bem, caso esteja interessado em ver o `html`, basta acessar o seguinte caminho na sua máquina virtual:

`cd /var/www/html`

```
lets@serverlets:~$ cd /var/www/html/
lets@serverlets:/var/www/html$ ls
index.html
lets@serverlets:/var/www/html$
```

Abra o arquivo com algum editor de texto de sua preferência. Por exemplo, pode-se aplicar:
`sudo nano index.html`

Na imagem abaixo, mostra-se uma alteração que foi realizada no conteúdo html. Para salvar as mudanças use CTRL+O. Use CTRL+X para sair do editor.



```
GNU nano 6.2 index.html *
}
color: #000000;
}

div.content_section_text a:hover {
background-color: #000000;

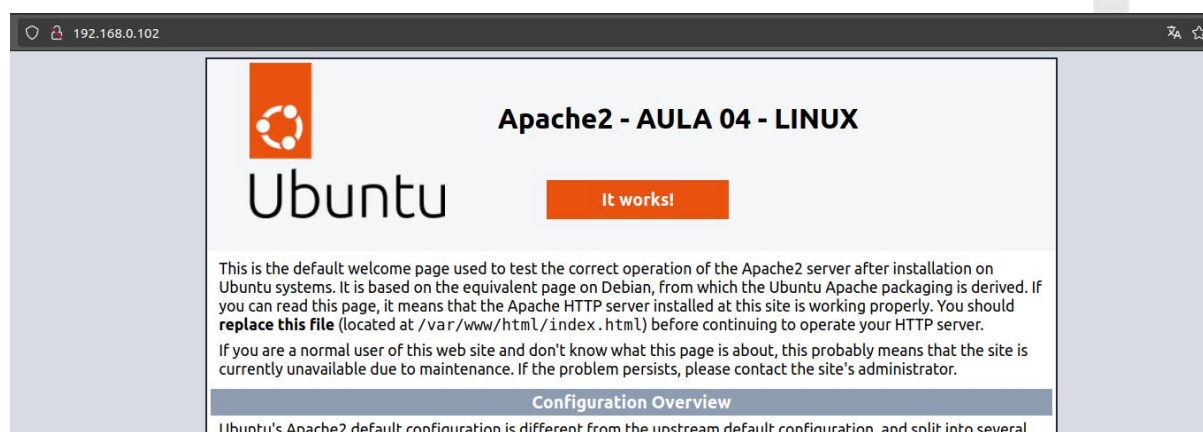
color: #DCDFE6;
}

div.validator {
}
</style>
</head>
<body>
<div class="main_page">
<div class="page_header floating_element">

</div>
<span style="margin-top: 1.5em;" class="floating_element">
Apache2 - AULA 04 - LINUX_
</span>
</div>
<div class="banner">
<div id="about"></div>
It works!
</div>

</div>
<div class="content_section floating_element">
<div class="content_section_text">
<p>
This is the default welcome page used to test the correct
Help Exit Write Out Read File Where Is Replace Cut Paste Execute Justify Location Go To Line M-U Undo M-E Redo
```

Recarregue a página web no navegador (F5) e notará as mudanças que você aplicou, conforme ilustrado abaixo:



Caso você não veja a mudança, pode ser necessário reiniciar o serviço do Apache. Nesse caso, aplique o comando abaixo:

sudo systemctl restart apache2

Parabéns, você acabou de servir a sua primeira página. Existe uma trilha exclusiva para servidores.

Compactação e descompactação

No Linux, a compactação e descompactação de arquivos são frequentemente realizadas através de utilitários de linha de comando. Existem várias extensões associadas a arquivos compactados, e cada uma delas indica o método de compressão utilizado. Os tipos mais comuns de arquivos são os **.tar**; **.tar.gz** e **.zip**, sendo que existem diferentes comandos para compactar e descompactar arquivos.

.tar

O formato de arquivo **.tar** é usado para agrupar vários arquivos e diretórios em um único arquivo, mas sem compressão (não se aplicou nenhum algoritmo de redução de conteúdo, trata-se apenas de um agrupamento). Portanto, o arquivo gerado pelo tar não é compactado e é frequentemente chamado de "tarball".

Para agrupar vários arquivos:

tar -cvf agrupado.tar arquivo1.txt arquivo2.txt arquivo3.txt

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
arquivo1.txt arquivo2.txt arquivo3.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tar -cvf agrupado.tar arquivo1.txt arquivo2.txt
arquivo3.txt
arquivo1.txt
arquivo2.txt
arquivo3.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
agrupado.tar arquivo1.txt arquivo2.txt arquivo3.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Para agrupar no diretório, basta escrever o nome do diretório:

tar -cvf agrupado.tar pasta/

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tree
.
├── pasta
│   ├── arquivo1.txt
│   ├── arquivo2.txt
│   └── arquivo3.txt
└── 1 directory, 3 files
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tar -cvf agrupado.tar pasta/
pasta/
pasta/arquivo1.txt
pasta/arquivo3.txt
pasta/arquivo2.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tree
.
├── agrupado.tar
├── pasta
│   ├── arquivo1.txt
│   ├── arquivo2.txt
│   └── arquivo3.txt
└── 1 directory, 4 files
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls -R
.:
agrupado.tar pasta
./pasta:
arquivo1.txt arquivo2.txt arquivo3.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

dentro do agrupado.tar tem um pasta/

Explicando as opções passadas:

- c:** Cria um novo arquivo.
- v:** Exibe o progresso. Essa flag vem da palavra verbose (verboso).
- f:** Especifica o nome do arquivo agrupado.

Podemos visualizar o conteúdo de um arquivo "tar", utilizando o comando:

tar -tf agrupado.tar

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
agrupado.tar
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tar -tf agrupado.tar
pasta/
pasta/arquivo1.txt
pasta/arquivo3.txt
pasta/arquivo2.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Explicando as opções:

- t: Exibe o conteúdo de um arquivo tar.
- f: Especifica o nome do arquivo agrupado.

Para desagrupar:

tar -xvf agrupado.tar

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
agrupado.tar
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tar -xvf agrupado.tar
pasta/
pasta/arquivo1.txt
pasta/arquivo3.txt
pasta/arquivo2.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
agrupado.tar pasta
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls -R
.:
agrupado.tar pasta

./pasta:
arquivo1.txt arquivo2.txt arquivo3.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Explicando as opções passadas:

- x: Extrair o conteúdo.
- v: Exibe o progresso. Essa flag vem da palavra verbose (verboso).
- f: Especifica o nome do arquivo agrupado.

.tar.gz

A principal função da ferramenta **tar** é consolidar vários arquivos em um único arquivo, mas, por si só, ela não reduz o tamanho do arquivo resultante, ou seja, não realiza compactação. Nesse contexto, entra em cena o **gzip** (GNU zip) ou outro compactador de sua preferência. Quando usado de forma independente, o gzip segue a seguinte sintaxe:

gzip [parâmetros] [nome-do-arquivo]

O arquivo compactado recebe uma extensão .gz.

Entre os parâmetros disponíveis, tem-se:

- c - extrai um arquivo para a saída padrão;
- d - descompacta um arquivo comprimido;
- l - lista o conteúdo de um arquivo compactado;
- v - exibe detalhes sobre o procedimento;
- r - compacta pastas;
- t - testa a integridade de um arquivo compactado.

Combinar as ferramentas **tar** e **gzip** permite agrupar e compactar. Basta adicionar o parâmetro **-z** ao usar a ferramenta tar. Por exemplo, para agrupar vários arquivos em um só, aplicando o algoritmo de compactação do gzip, aplique o seguinte comando:

tar -zcvf trilhas.tar.gz

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
css.txt  html.txt  js.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tar -zcvf trilhas.tar.gz css.txt html.txt js.txt
css.txt
html.txt
js.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
css.txt  html.txt  js.txt  trilhas.tar.gz
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tar -tf trilhas.tar.gz
css.txt
html.txt
js.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Para descompactar um arquivo compactado .tar.gz basta utilizar:

tar -zxvf trilhas.tar.gz

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
trilhas.tar.gz
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tar -zxvf trilhas.tar.gz
css.txt
html.txt
js.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
css.txt  html.txt  js.txt  trilhas.tar.gz
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

.zip

O formato de arquivo compactado .zip é amplamente utilizado para comprimir arquivos e diretórios no mundo da computação. O utilitário de linha de comando comum para trabalhar com ele é o zip.

Para compactar arquivos, utilize:

zip compactado.zip arquivo1.txt arquivo2.txt arquivo3.txt

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
css.txt  html.txt  js.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ zip compactado.zip css.txt html.txt js.txt
  adding: css.txt (stored 0%)
  adding: js.txt (stored 0%)
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
compactado.zip  css.txt  html.txt  js.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

O formato Para compactar um diretório use a flag -r:

zip -r compactado.zip pasta/

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls -R
.:
pasta

./pasta:
css.txt  html.txt  js.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ zip -r compactado.zip pasta/
  adding: pasta/ (stored 0%)
  adding: pasta/js.txt (stored 0%)
  adding: pasta/html.txt (stored 0%)
  adding: pasta/css.txt (stored 0%)
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
compactado.zip  pasta
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ unzip -l compactado.zip
Archive:  compactado.zip
  Length   Date   Time    Name
  -----  -
      0  2024-02-04 17:08   pasta/
     21  2024-02-04 16:45  pasta/js.txt
     15  2024-02-04 16:45  pasta/html.txt
     14  2024-02-04 16:45  pasta/css.txt
  -----  -
     50                      4 files
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Para visualizar o conteúdo de um arquivo compactado zip, utilize unzip com a flag -l (L minúsculo):
unzip -l compactado.zip

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
compactado.zip
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ unzip -l compactado.zip
Archive: compactado.zip
  Length      Date    Time    Name
  -----
    14      2024-02-04  16:45    css.txt
    15      2024-02-04  16:45    html.txt
    21      2024-02-04  16:45    js.txt
  -----
    50                      3 files
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Para descompactar, basta usar o unzip:

unzip compactado.zip

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
compactado.zip
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ unzip compactado.zip
Archive: compactado.zip
  extracting: css.txt
  extracting: html.txt
  extracting: js.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
compactado.zip  css.txt  html.txt  js.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Busca de arquivos

Se você está iniciando no uso do Linux, pode não estar familiarizado com a busca de arquivos e pastas em seu computador. Existem dois comandos no Linux projetados para realizar essa tarefa: **find** e **locate**.

No ambiente Linux, o comando **find** é uma ferramenta versátil utilizada para localizar arquivos e diretórios, oferecendo opções como pesquisa por nome, tipo e tamanho, com resultados exibidos em tempo real. Por outro lado, o comando **locate** é uma opção mais ágil que depende de um banco de dados pré-existente para buscas rápidas de arquivos, embora possa não refletir atualizações em tempo real.

Comando find

O comando mais amplamente utilizado para buscar e filtrar arquivos no ambiente Linux é o find. Sua sintaxe básica é a seguinte:

find <diretório-inicial> <opções> <termo-de-pesquisa>

O argumento **<diretório-inicial>** representa o ponto de origem a partir do qual você deseja iniciar a busca. Este argumento pode ser substituído por:

- **/** (barra): pesquise todo o sistema.
- **.** (ponto): pesquise a partir da pasta em que você está trabalhando atualmente (diretório atual).
- **~** (til): para pesquisar a partir da sua pasta inicial.

O segundo parâmetro, **<opções>**, é destinado às características do arquivo, como o nome, tipo, data de criação, entre outros. O terceiro parâmetro, **<termo de pesquisa>**, é o espaço onde você especifica o termo relevante para a busca.

Pesquisa pelo nome

Para executar uma consulta de pesquisa simples usando o nome do arquivo, use a flag **-name**:
find . -name <nome-do-arquivo>

Essa flag é "case sensitive", ou seja, ocorre diferenciação entre minúsculas e maiúsculas. Se você sabe o nome do arquivo, mas não tem certeza se ele usa maiúsculas ou minúsculas, use a flag **-iname**, ela não diferencia maiúsculas de minúsculas, conforme apresentado a seguir:

find . -iname <nome-do-arquivo>

A imagem abaixo exemplifica a aplicação de ambas as opções. É essencial destacar a função do asterisco (*) como um coringa. Ao empregá-lo, torna-se dispensável especificar a extensão do arquivo. A imagem abaixo mostra alguns exemplos de aplicação. Note que um deles usa -name, sem informar corretamente o nome do arquivo. Consequentemente, a ferramenta não retorna resultados. Em contrapartida, no caso da existência do arquivo, ele retorna o caminho até o arquivo.

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tree
.
├── a-pasta
│   ├── css.txt
│   ├── html.txt
│   ├── Index.txt
│   └── js.txt
├── compactado.zip
├── outra-pasta
│   ├── dados.json
│   └── Lista.txt
└── 2 directories, 7 files

letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ find . -name "html*"
./a-pasta/html.txt

letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ find . -name "Html*"

letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ find . -iname "Html*"
./a-pasta/html.txt

letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

É possível procurar por vários arquivos com um formato em comum, como .txt, .csv, etc., através desse comando:

find . -name "*.txt"

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tree
.
├── a-pasta
│   ├── css.txt
│   ├── html.txt
│   ├── Index.txt
│   └── js.txt
├── compactado.zip
├── outra-pasta
│   ├── dados.json
│   └── Lista.txt
└── 2 directories, 7 files

letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ find . -name "*.txt"
./a-pasta/js.txt
./a-pasta/html.txt
./a-pasta/Index.txt
./a-pasta/css.txt
./outra-pasta/Lista.txt

letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

É possível procurar por todos os arquivos que não contêm uma palavra-chave específica em seu nome. Isso pode ser feito da seguinte forma:

find . -not -name <termo-de-pesquisa>
find . \! -name <termo-de-pesquisa>

Na imagem abaixo, dentro da pasta temos dois arquivos. E fizemos uma pesquisa retornando os arquivos que não tem "*.txt", logo, retorna dados.json.

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ ls
dados.json  Lista.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ find . -not -name "*.txt"
.
./dados.json
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $
```

É possível encontrar um determinado arquivo pelo nome e removê-lo, use o argumento `-delete` após o nome do arquivo (termo de pesquisa):

`find . -iname <termo-de-pesquisa> -delete`

Na imagem abaixo, optamos por usar `-iname`, para que não fosse necessário saber exatamente se o arquivo usa letras maiúsculas e minúsculas no nome.

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls -R
.:
a-pasta  compactado.zip  outra-pasta

./a-pasta:
css.txt  html.txt  Index.txt  js.txt

./outra-pasta:
dados.json  Lista.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ find . -iname "INDEX*" -delete
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls -R
.:
a-pasta  compactado.zip  outra-pasta

./a-pasta:
css.txt  html.txt  js.txt

./outra-pasta:
dados.json  Lista.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Pesquisa pelo tipo

No Linux, os usuários têm a capacidade de listar informações com base nos tipos de arquivo. Diversos filtros estão disponíveis para essa finalidade:

- d:** Representa diretórios.
- f:** Indica arquivos normais.
- l:** Refere-se a links simbólicos.
- c:** Denota dispositivos de caractere.
- b:** Corresponde a dispositivos de bloco.

Por exemplo, o comando a seguir permite encontrar todas as pastas a partir do diretório atual (.):

`find . -type d`

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ ls
a-pasta  compactado.zip  outra-pasta
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ find . -type d
.
./a-pasta
./outra-pasta
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

É possível combinar **-type** com **-name**. O exemplo abaixo procuraria apenas por arquivos (files), excluindo diretórios e links:

```
find . -type f -name <meu-arquivo>
```

Pesquisa por data e hora

Podemos procurar arquivos com base no horário de acesso ou modificação. O Linux mantém o controle dos arquivos usando esses três carimbos de data/hora:

- Horário de Acesso ou **access time**, em inglês (-atime): quando o arquivo foi lido ou escrito.
- Horário de Modificação ou **modification time**, em inglês (-mtime): quando o arquivo foi modificado.
- Horário da Alteração ou **change time**, em inglês (-ctime): quando os metadados do arquivo foram atualizados.

Devemos empregar uma das opções (-atime, -mtime, -ctime) junto com um número que especifica quantos dias se passaram desde que o arquivo foi acessado, modificado ou alterado.

O exemplo a seguir retorna todos os arquivos que foram acessados a menos de um dia, a partir do seu horário atual.

```
find . -atime -1
```

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ find . -atime -1
.
./a-pasta
./a-pasta/js.txt
./a-pasta/html.txt
./a-pasta/css.txt
./outra-pasta
./outra-pasta/dados.json
./outra-pasta/Lista.txt
./compactado.zip
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Podemos refinar ainda mais nossas consultas adicionando sinais de **mais** (+) e **menos** (-) precedendo o número de dias. No exemplo acima, o sinal de menos (-) refinou a pesquisa para retornar todos os arquivos/pastas que foram acessados a menos de um dia (-1).

No exemplo abaixo, +2 indica que estamos procurando os arquivos que foram modificados a mais de 2 dias.

```
find ~ -mtime +2
```

Pesquisa por tamanho

O Linux permite que você procure arquivos com base em seus tamanhos. A sintaxe para procurar arquivos por tamanho é:

```
find <diretório-inicial> -size <tamanho> <unidade-de-tamanho>
```

O elemento **<unidade-de-tamanho>** pode especificar as seguintes unidades de tamanho:

- c - bytes
- k - quilobytes
- M - megabytes
- G - gigabytes
- b - pedaços de 512 bytes

Se não for informado nenhum sinal (+ ou -), a busca procura arquivos com exatamente aquela quantidade de bytes. O funcionamento do sinal é semelhante ao que foi explicado para datas. Na imagem abaixo, mostra-se 3 exemplos: procurar arquivos com exatamente 71k, arquivos com menos de 4k e, por fim, arquivos com mais de 8k.

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/a-pasta $ ls -lh
total 88K
-rw-rw-r-- 1 letonio.silva letonio.silva 71K fev  4 20:39 css.txt
-rw-rw-r-- 1 letonio.silva letonio.silva 9,3K fev  4 20:40 html.txt
-rw-rw-r-- 1 letonio.silva letonio.silva 1,6K fev  4 20:44 js.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/a-pasta $ find . -size 71k
./css.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/a-pasta $ find . -size -4k
./js.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/a-pasta $ find . -size +8k
./html.txt
./css.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/a-pasta $
```

Comando locate

O comando **locate** é uma opção conveniente, pois é mais ágil que o comando **find** ao realizar buscas. Isso se deve ao fato de que o **locate** verifica apenas o banco de dados Linux, em vez de percorrer todo o sistema. Além disso, sua sintaxe é relativamente mais simples de escrever, como mostrado a seguir:

locate <nome-do-arquivo>

Às vezes, o comando **locate** padrão pode retornar arquivos que foram excluídos, se o banco de dados não foi atualizado. A melhor solução é atualizar manualmente o banco de dados executando o seguinte:

sudo updatedb

Por padrão, o Linux não vem com o comando **locate** pré-instalado. Nesse caso, faça a instalação usando **sudo apt install locate**

A imagem a seguir mostra a busca pelo arquivo **css.txt**. Além disso, nota-se que ele mostra o caminho absoluto.

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ tree
.
├── a-pasta
│   ├── css.txt
│   ├── html.txt
│   └── js.txt
├── compactado.zip
├── outra-pasta
│   ├── dados.json
│   └── Lista.txt
└── 2 directories, 6 files
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ locate "css.txt"
/home/letonio.silva/Documentos/alphaedtech/examples/a-pasta/css.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

A sintaxe básica só permite que você procure por arquivos que contêm o termo de pesquisa. Se você quiser obter um arquivo com o seu nome exato, pode usar a opção `-r` e adicionar o símbolo do cifrão (\$) no final do seu termo de pesquisa. Por exemplo:

locate -r nome-do-arquivo\$

Para saber quantos arquivos aparecem no resultado da sua pesquisa, insira `-c` após o comando `locate`.

locate -c <nome-do-arquivo>

Em vez de listar todos os arquivos, ele fornecerá a quantidade total de arquivos que batem com o termo.

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ locate -c mysql
448
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Use `-i` no seu comando `locate` para ignorar a diferença entre maiúsculas e minúsculas na busca. Por exemplo:

locate -i <nome-do-arquivo>

Se você quer limitar o número de resultados da pesquisa, use a opção `-l <número>` (ou `--limit`). No entanto, lembre-se de que você precisa colocar a opção no final da linha de comando. Dê uma olhada neste exemplo:

locate <nome-do-arquivo> -l 5

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $ locate *.txt -l 5
/etc/X11/rgb.txt
/etc/brltty/Input/ba/all.txt
/etc/brltty/Input/bd/all.txt
/etc/brltty/Input/bl/18.txt
/etc/brltty/Input/bl/40_m20_m40.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples $
```

Comando grep

A pesquisa em arquivos de texto é uma operação crucial, especialmente ao lidar com a edição de arquivos de configuração de serviços. Uma ferramenta fundamental para essa tarefa no Unix é o comando **grep (Global Regular Expression Print)**. Essa ferramenta versátil e útil permite que os usuários busquem padrões específicos dentro de arquivos de texto. Em essência, o **grep** possibilita a pesquisa por palavras ou padrões, exibindo as linhas correspondentes ou, se necessário, múltiplas linhas. A sintaxe do Comando Grep ao realizar uma busca fica assim:

grep <opções> padrão <arquivo>

O comando oferece muitas possibilidades, mas as mais utilizadas são:

- i a pesquisa não irá diferenciar letras maiúsculas de minúsculas. Ou seja, se você pesquisar por "carro" será o mesmo que pesquisar "CARRO".

- c irá mostrar somente o número de linhas que combinam com o padrão pesquisado.

- r habilita pesquisa recursiva no diretório atual.

- v com esta opção, as linhas que não combinam com o padrão pesquisado serão exibidas.

Localizar palavra-chave em um arquivo, por exemplo:

grep palavra-chave <arquivo>

Na imagem a seguir, procura-se pela palavra "utilizado"

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ ls
dados.json frontend.txt Lista.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ grep utilizado frontend.txt
CSS é utilizado para estilizar documentos HTML. Ele fornece regras de estilo que determinam a
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $
```


Para ignorar letras minúsculas ou maiúscula no padrão procurado, use a flag -i, conforme apresentado a seguir:

grep -i palavra-chave <arquivo>

```
letonio.silva@BRRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ grep -i ELE frontend.txt
Ele define elementos como títulos, parágrafos, links, imagens e outros, permitindo a
CSS é utilizado para estilizar documentos HTML. Ele fornece regras de estilo que determinam a
interatividade e dinamismo às páginas web. Ele permite a manipulação do DOM (Document Object
Model) para alterar elementos na página, responder a eventos do usuário e realizar operações
letonio.silva@BRRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $
```

Com a opção -c é possível descobrir quantas vezes a palavra pesquisada aparece no arquivo de texto.

grep -c palavra-chave <arquivo>

Na imagem abaixo, usamos duas opções (-c e -i) para determinar quantas vezes o padrão "ELE" aparece, independente se é formado por letras maiúsculas ou minúsculas.

```
letonio.silva@BRRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ grep -i ELE frontend.txt
Ele define elementos como títulos, parágrafos, links, imagens e outros, permitindo a
CSS é utilizado para estilizar documentos HTML. Ele fornece regras de estilo que determinam a
interatividade e dinamismo às páginas web. Ele permite a manipulação do DOM (Document Object
Model) para alterar elementos na página, responder a eventos do usuário e realizar operações
letonio.silva@BRRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ grep -c -i ELE frontend.txt
4
letonio.silva@BRRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $
```

Para pesquisar várias palavras podemos fazer assim:

grep -E "padrao1 | padrao2" arquivo.txt

```
letonio.silva@BRRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ grep -E "CSS|JS" frontend.txt
CSS (Cascading Style Sheets):
CSS é utilizado para estilizar documentos HTML. Ele fornece regras de estilo que determinam a
estrutura do conteúdo da sua apresentação, CSS oferece flexibilidade e facilita a manutenção do
JavaScript (JS):
letonio.silva@BRRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ grep -c -E "CSS|JS" frontend.txt
4
letonio.silva@BRRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $
```

Para pesquisar de forma recursiva dentro de um diretório, ideal para pesquisas em grandes conjuntos de arquivos, use:

grep -r palavra-chave /caminho/do/diretorio

Para excluir todas as linhas que contêm a palavra-chave no arquivo, use a opção -v. Ela é útil para filtrar linhas indesejadas. Na imagem a seguir, queremos todas as linhas que não têm "de".

```
letonio.silva@BRRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ cat frontend.txt
HTML (Hypertext Markup Language):
HTML é a linguagem de marcação utilizada para estruturar o conteúdo de uma página web.
Ele define elementos como títulos, parágrafos, links, imagens e outros, permitindo a
organização lógica e hierárquica do conteúdo.

CSS (Cascading Style Sheets):
CSS é utilizado para estilizar documentos HTML. Ele fornece regras de estilo que determinam a
apresentação visual do conteúdo, incluindo layout, cores, fontes e espaçamento. Ao separar a
estrutura do conteúdo da sua apresentação, CSS oferece flexibilidade e facilita a manutenção do
design em páginas web.

JavaScript (JS):
JavaScript é uma linguagem de programação de alto nível, frequentemente usada para adicionar
interatividade e dinamismo às páginas web. Ele permite a manipulação do DOM (Document Object
Model) para alterar elementos na página, responder a eventos do usuário e realizar operações
assíncronas, como chamadas de API.
letonio.silva@BRRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ grep -v "de" frontend.txt
HTML (Hypertext Markup Language):
organização lógica e hierárquica do conteúdo.

CSS (Cascading Style Sheets):
apresentação visual do conteúdo, incluindo layout, cores, fontes e espaçamento. Ao separar a

JavaScript (JS):
letonio.silva@BRRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $
```


Também é possível pesquisar em diversos arquivos por uma palavra em um único comando:

grep -l solicitação ./*

Na próxima seção vamos aprender piping, que permite encadear comandos em sequência, de forma que a saída de um comando é enviada como entrada no próximo comando. É comum combinar outros comandos com o grep, permitindo a busca em tempo real em dados gerados por outros comandos. A sintaxe é a seguinte:

comando | grep palavra-chave

Piping

Piping é um conceito fundamental em sistemas Linux, e refere-se à capacidade de encadear a saída de um comando como entrada para outro comando. O símbolo utilizado para indicar o piping é o caractere de barra vertical (|).

Quando você utiliza o piping, o output (saída) de um comando é direcionado como input (entrada) para outro comando subsequente. Isso permite a criação de sequências de comandos complexas e poderosas, aproveitando a modularidade e a especialização de diferentes programas.

A sintaxe básica é:

comando1 | comando2

A seguir, veremos alguns exemplos de aplicação.

Filtragem de resultados

Na imagem abaixo, a listagem detalhada dos arquivos (ls -l) é filtrada para mostrar apenas as linhas que contenham a palavra "txt" usando o comando grep.

ls -l | grep "txt"

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ ls
dados.json frontend.txt Lista.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ ls -l | grep "txt"
-rw-rw-r-- 1 letonio.silva letonio.silva 960 fev  4 21:46 frontend.txt
-rw-rw-r-- 1 letonio.silva letonio.silva  54 fev  4 18:39 Lista.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $
```

Contagem de linhas

O conteúdo do arquivo é passado para o comando wc -l (wordcount com flag para linhas), que conta o número de linhas. Na última linha, conta-se a quantidade de palavras (flag -w).

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ ls
dados.json frontend.txt Lista.txt
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ cat frontend.txt
HTML (Hypertext Markup Language):
HTML é a linguagem de marcação utilizada para estruturar o conteúdo de uma página web.
Ele define elementos como títulos, parágrafos, links, imagens e outros, permitindo a
organização lógica e hierárquica do conteúdo.

CSS (Cascading Style Sheets):
CSS é utilizado para estilizar documentos HTML. Ele fornece regras de estilo que determinam a
apresentação visual do conteúdo, incluindo layout, cores, fontes e espaçamento. Ao separar a
estrutura do conteúdo da sua apresentação, CSS oferece flexibilidade e facilita a manutenção do
design em páginas web.

JavaScript (JS):
JavaScript é uma linguagem de programação de alto nível, frequentemente usada para adicionar
interatividade e dinamismo às páginas web. Ele permite a manipulação do DOM (Document Object
Model) para alterar elementos na página, responder a eventos do usuário e realizar operações
assíncronas, como chamadas de API.
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ cat frontend.txt | wc -l
16
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ cat frontend.txt | wc -w
135
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $
```

Ordenação

A lista é ordenada alfabeticamente utilizando o comando **sort**.
cat <arquivo> | sort

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ cat Lista.txt
Sofá
Cama
Mesa de jantar
Cadeiras
Armários
Estantes
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ cat Lista.txt | sort
Armários
Cadeiras
Cama
Estantes
Mesa de jantar
Sofá
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $
```

No exemplo a seguir, echo por si só imprimiria a mensagem no terminal. Em vez disso, o conteúdo é enviado como entrada para o comando **tr** (translate ou delete characters), que converte letras minúsculas em maiúsculas.

```
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $ echo "Olá, mundo!" | tr '[:lower:]' '[:upper:]'
OLÁ, MUNDO!
letonio.silva@BRRIOLN043879:~/Documentos/alphaedtech/examples/outra-pasta $
```

Você pode encontrar mais exemplos de piping aqui:

<https://www.geeksforgeeks.org/piping-in-unix-or-linux/>