

PROGRAMAÇÃO EM SHELL SCRIPT

UNIDADE 02 - COMANDOS LINUX

Emmanuel Joca Nolêto

Introdução

Olá, estudante! Seja bem-vindo a esta unidade de **Programação em Shell Script**. Iniciaremos nosso estudo com a apresentação dos comandos que são utilizados nos sistemas operacionais Linux. Esse conhecimento nos dará uma maior amplitude de operações dentro do sistema.

Você já conhece os comandos em Shell Script? Sabe como identificar erros em um código em Shell? Então, fique atento, pois essas indagações serão respondidas ao longo das próximas páginas.

Como você já deve saber, há uma diversidade de comandos no Shell Script, e eles ficam armazenados dentro de um arquivo, sendo executados sempre que houver necessidade. Vale ressaltar que esses comandos são os mesmos executados pelo terminal de comando, ou seja, tudo é feito por meio do terminal Shell. Uma recomendação é evitar, num primeiro momento, a utilização do modo terminal por meio do usuário *root* de seu computador. Essa precaução ocorre devido à responsabilidade que esse usuário tem no momento de executar um comando, ou um conjunto de comandos. Por ser o superusuário do sistema Linux, ele vai executar todo e qualquer comando, caso sua sintaxe esteja correta, sem se preocupar com as potenciais consequências. Dependendo do comando ou sequência, dá pra prever que danos irreversíveis poderão ser causados ao sistema operacional, certo? Situação que só será corrigida com uma nova instalação do sistema.

Toda essa precaução é essencial para usuários iniciantes, porém, tenha em mente que essa é apenas uma fase. Com o aprofundamento dos estudos, mediante a leitura e a prática dos comandos que estão contidos neste material, você terá maior confiança e poderá desfrutar das possibilidades de um superusuário no computador.

Ao finalizar esta unidade, você dominará alguns conceitos que, nos aspectos de comandos Shell Script, serão cruciais para sua vida profissional na área de tecnologia.

Siga em frente e estude esta unidade com bastante atenção!

2.1 Comandos de gerenciamento de pacotes

Os sistemas operacionais Linux permitem que sejam realizados todos os seus procedimentos por meio de comandos em modo Shell. Esses comandos podem ser utilizados para a configuração e instalação de programas, ou *softwares*, e ainda permitem que seja feita a administração dos pacotes de instalação para que as dependências a serem instaladas estejam de acordo com as necessidades do usuário. Saiba mais a seguir!

2.1.1 Pacotes e dependências

O gerenciamento de pacotes é feito para atender a necessidade de o sistema ter controle sobre a atualização, a remoção e a instalação de aplicativos.

Clique nas abas abaixo e aprenda mais sobre o tema.

Esse gerenciamento pode ser feito por meio de aplicações que darão acesso ao sistema operacional Linux por meio de um *software* com *layout* visual ou pelo terminal Shell, cujos comandos permitem fazer o mesmo e de forma mais eficiente, por não se prenderem ao aspecto visual, mas, sim, à execução da tarefa.

O primeiro comando que você deve aprender para gerenciar pacotes é o APT (*Advanced Packaging Tool*). Esse comando pode ser utilizado como *“apt update”*, que realiza a atualização da lista de pacotes, e *“apt upgrade”*, que realiza a atualização de todo o sistema operacional. Outro emprego do comando APT é a atualização dos repositórios de pacotes, responsáveis por disponibilizar a versão homologada dos pacotes que se deseja baixar, ou seja, a versão mais estável.

Esse comando também pode ser utilizado para adicionar algum pacote de um repositório externo. Por meio do comando “*apt-add-repository*”, é possível inserir no sistema um novo repositório e baixar o que se deseja dele. No caso de o usuário querer um *software* que não está disponível nos repositórios do próprio sistema operacional, esse comando permitirá a ele utilizá-lo de outro local, bastando apenas saber o repositório.

VOCÊ SABIA?



Há alguns programas, denominados “*front-ends*”, que são desenvolvidos para a listagem dos pacotes disponíveis para instalação e atualização de *software* no sistema operacional Linux. Saiba mais detalhes em: <https://www.debian.org/doc/manuals/apt-howto/ch-search.pt-br.html>.

A diversidade de possibilidades de comandos para um gerenciamento de pacotes por meio do APT é bem ampla. A seguir, temos uma imagem que representa uma pequena lista de comandos que o terminal exibe:

```
noletto@tanatos: ~  
Arquivo Editar Ver Pesquisar Terminal Ajuda  
noletto@tanatos:~$ apt  
apt  
apt-add-repository  
apt-cache  
apt-cdrom  
apt-config  
aptd  
aptdcon  
apt-extracttemplates  
apt-ftarchive  
apt-get  
apt-key  
apt-mark  
apt-sortpkgs  
apturl  
apturl-gtk  
noletto@tanatos:~$
```

Figura 1 - Lista de opções do comando APT.

Fonte: Elaborada pelo autor, 2019.

Conhecer esses comandos possibilita ao usuário criar seu próprio comando automatizado para realizar o processo de atualização. Dependendo do grau de maturidade, o usuário pode fazer um script em Shell que venha a atualizar vários computadores em uma rede, e inclusive os servidores. A programação em Shell Script abre um grande leque de possibilidades que podem ser criadas apenas usando comandos básicos de entrada e saída de dados, comandos de repetição e condição atrelados aos comandos Shell do próprio sistema Linux.

2.1.2 YUM versus APT

Tenha em mente, contudo, que o Linux dispõe de diversas maneiras de realizar a instalação de pacotes. Existem muitos tipos de instaladores para realizar essa tarefa, porém, é claro, há diferenças entre eles. Há, por exemplo, distribuições que são baseadas na distribuição Debian, utilizando, como já vimos, o instalador APT.

Clique nas abas, para aprender mais sobre o tema.

APT

De fato, o APT é o mais conhecido para essa distribuição e para as distribuições que são oriundas dela. Sua capacidade de resolver as dependências é essencial para sua aceitação pelos usuários do sistema. Seu trabalho ocorre em conjunto com o DPKG, outra ferramenta de gerenciamento de pacotes, mas ela realiza o gerenciamento por meio de códigos de

	baixo nível, fazendo com que haja integridade entre as dependências dos pacotes. Saiba que quando o DPKG e o APT trabalham juntos o resultado é um processo de instalação e remoção mais amigável para o usuário.
Uso do APT	O uso do APT pelas distribuições Debian, ou distribuições dela oriundas, é algo que permite a integridade entre as dependências. Vale dizer que sua simplicidade na sintaxe é um atrativo para quem deseja aprender como os sistemas Linux funcionam.
Atualmente	Atualmente, os comandos do APT estão sendo resumidos ou reescritos para que se tornem ainda mais fáceis e amigáveis. Infelizmente, ainda existem algumas limitações, como o fato de não se poder instalar um pacote de uma URL, o que faz com que o usuário tenha que realizar todo o procedimento de outra forma, ou seja, usando o DPKG.

Outro método similar ao APT é o YUM, que trabalha com distribuições que utilizam extensões RPM. Seu emprego é recorrente por parte de distribuições Linux.

VOCÊ QUER LER?



Recomendamos a leitura do artigo “Gerenciamento de pacote RPM e YUM”, da IBM, que você pode encontrar no link: <https://www.ibm.com/developerworks/br/linux/library/l-lpic1-v3-102-5/index.html>. O texto proporcionará um maior aprofundamento dos conhecimentos dessa outra forma de trabalhar com o sistema Linux.

Tenha em mente que distribuições como Fedora e CentOS utilizam o YUM. A simplicidade de sua linguagem e a proximidade com os comandos do APT possibilitam que um usuário utilize os comandos do YUM de forma facilitada. O YUM foi criado para atender apenas a uma distribuição denominada “YellowDog”, contudo, seu desempenho para Shell Script era tão satisfatório que logo foi percebido por outros usuários do sistema operacional Linux. Isso despertou o interesse de ver o YUM funcionando nas demais distribuições. Concomitantemente, na época da sua criação, a distribuição RedHat/Fedora estava precisando de um *software* que permitisse que suas atualizações fossem feitas e que resolvesse os problemas em relação à dependência de pacotes.

As particularidades dos comandos para remover dependências, ou para correlacioná-las, são muitas, e isso é um dos grandes desafios do sistema operacional Linux. Sua forma de escrita é diferenciada quando se trata de algumas distribuições, causando uma incompatibilidade. Para suprir a lacuna que há no comando de dependências, temos o comando ZYPP, porém ele deve ser instalado para ser utilizado. As principais distribuições que o utilizam são:

- Red Hat Enterprise Linux (RHEL);
- Fedora;
- CentOS;
- Mandriva;
- OpenSuSE;
- Scientific Linux;
- Oracle Linux;
- ClearOS;
- ROSA Enterprise Linux Server;

- Rocks Cluster Distribution;
- Fermi Linux.

Clique nas setas e conheça mais sobre o tema.

Esse gerenciador de arquivos contém uma grande amplitude de recursos e de repositórios, mas muitos desses repositórios são de terceiros, e, por isso, podem conter também outros tipos de repositórios não oficiais, dos quais não se sabe a origem. Não é recomendado fazer a integração com os scripts em Shell que já se encontram no sistema do computador.

Outra opção de instalador de pacotes é o APT, também conhecido por “apt-get”, comumente utilizado por usuários dos sistemas operacionais Linux. Esse instalador é responsável por manipular pacotes de maneira similar ao YUM, sendo considerado uma versão aperfeiçoada do DPKG, um antigo gerenciador de pacotes que ainda existe. No APT-GET, existe um arquivo de configuração que fica localizado no endereço `/etc/apt/sources.list`, que contém uma lista de todos os servidores que podem ter pacotes para *download* de programas e atualizações.

Alguns dos itens dessa lista são de repositórios oficiais da Debian — como você já deve saber, uma distribuição Linux muito bem-conceituada — assim como de repositórios não oficiais, os quais podem ter pacotes não oficiais. No momento em que o comando `APT-GET UPDATE`, por exemplo, é executado em seu terminal Shell, será feita uma busca de todos os servidores que estiverem disponíveis, de acordo com a lista dos acessos que está no arquivo `sources.list`. Essa execução fará o *download* de uma espécie de base de dados compactada, que virá de cada um dos servidores listados no arquivo anteriormente citado. Essa base permitirá o acesso a pacotes, versões que possam existir, entre outras características que o sistema possa precisar para seu perfeito funcionamento. Frequentemente, são dependências que os *softwares* precisam e, apenas por meio do comando `APT-GET UPDATE`, serão verificadas e atualizadas.

Quando é terminado o *download* da base, o sistema operacional, por meio de um script, realiza a junção das bases menores em uma base única, a fim de garantir tudo o que precisa ser feito. O *download*, portanto, estará centralizado em apenas um arquivo ou base. Ao finalizar essa operação de atualização, o usuário do sistema tem a possibilidade de fazer a instalação de todo e qualquer *software* que esteja na base por meio do comando `APT-GET INSTALL`, que irá verificar se o programa desejado está listado.

Atualmente, existe uma nova forma de usar o comando `APT-GET` por meio do `APT`, que veio para resolver o problema de dois comandos de gerenciamento: o `APT-GET` e o `APT-CACHE`. Ele é considerado a junção desses dois comandos, evitando, assim, confusões que existiam sobre o que cada um fazia, especialmente ao constatar a similaridade dos resultados que cada um produz.

Com o comando `APT`, foi feita uma espécie de centralização das informações em um só lugar e uma redução de comandos, conforme já citado anteriormente. Essa forma resumida teve uma excelente aceitação por parte dos usuários do sistema operacional Linux. Um dos recursos criados com o novo comando foi uma barra de progresso para que o usuário tenha noção de tudo que está sendo feito e de como está o andamento de forma ilustrada, ou seja, ele permite a visualização do número de pacotes que podem ser atualizados no momento em que está sendo feita essa atualização referente ao banco de repositórios.

Com todos esses novos recursos, o comando `APT-GET` não pode ser considerado obsoleto devido às suas múltiplas funcionalidades; todas consolidadas pela comunidade de usuários. Não existe nenhuma informação a respeito da descontinuidade do comando e a troca do comando `APT-GET` pelo comando `APT` é bem confortável para quem deseja mais praticidade e velocidade nos *downloads* de programas, atualizações e dependências do sistema em uso.

2.2 Comandos de gerenciamento de processos em Shell

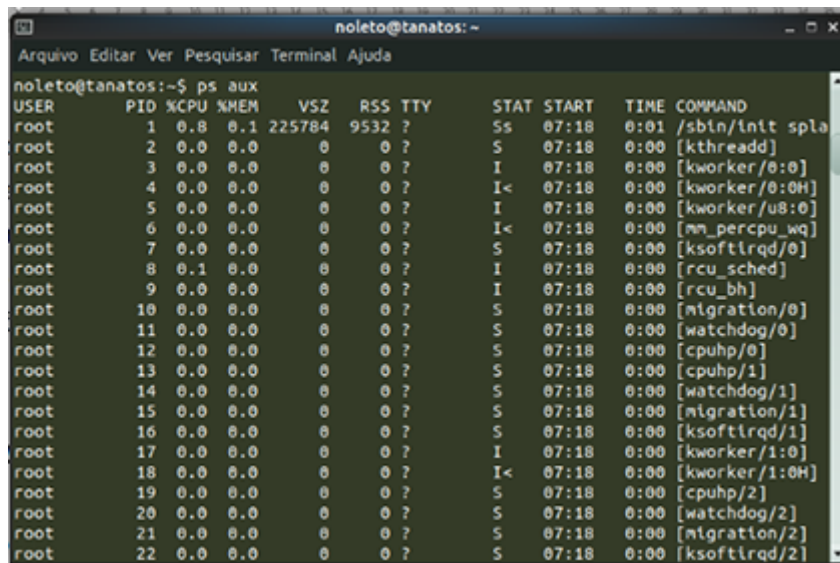
Todos os sistemas operacionais têm uma gama de processos que ficam sendo executados constantemente para atender as demandas solicitadas. É nessa execução de várias tarefas simultaneamente que podemos verificar o quão rápido é o sistema. Diante disso, podemos constatar que os sistemas têm um perfeito sincronismo de suas tarefas e, por meio desse sincronismo, os processos podem ser executados.

2.2.1 Processos de um sistema Linux

Aqui, vamos aprender como se dá o funcionamento dos comandos que realizam o gerenciamento por meio de script feito em Shell. Uma informação essencial é que muitos processos compartilham a mesma biblioteca, que está alocada em memória por meio de seu código. O mais interessante é que, quando for necessária alguma pequena modificação nessa biblioteca, mesmo que ela esteja em memória, a ação pode ser realizada.

Antes de mais nada, devemos saber o que é um processo. Segundo Tanenbaum e Woodhull (2008, p. 71),

[...] a ideia-chave aqui é que um processo é um tipo de atividade. Ele tem um programa, entrada, saída e um estado. Um único processador pode ser compartilhado entre vários processos, com algum algoritmo de agendamento sendo utilizado para determinar quando parar de trabalhar em um processo e servir a um diferente.



USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.1	225784	9532	?	Ss	07:18	0:01	/sbin/init splash
root	2	0.0	0.0	0	0	?	S	07:18	0:00	[kthreadd]
root	3	0.0	0.0	0	0	?	I	07:18	0:00	[kworker/0:0]
root	4	0.0	0.0	0	0	?	I<	07:18	0:00	[kworker/0:0H]
root	5	0.0	0.0	0	0	?	I	07:18	0:00	[kworker/u8:0]
root	6	0.0	0.0	0	0	?	I<	07:18	0:00	[mm_percpu_wq]
root	7	0.0	0.0	0	0	?	S	07:18	0:00	[ksoftirqd/0]
root	8	0.1	0.0	0	0	?	I	07:18	0:00	[rcu_sched]
root	9	0.0	0.0	0	0	?	I	07:18	0:00	[rcu_bh]
root	10	0.0	0.0	0	0	?	S	07:18	0:00	[migration/0]
root	11	0.0	0.0	0	0	?	S	07:18	0:00	[watchdog/0]
root	12	0.0	0.0	0	0	?	S	07:18	0:00	[cpuhp/0]
root	13	0.0	0.0	0	0	?	S	07:18	0:00	[cpuhp/1]
root	14	0.0	0.0	0	0	?	S	07:18	0:00	[watchdog/1]
root	15	0.0	0.0	0	0	?	S	07:18	0:00	[migration/1]
root	16	0.0	0.0	0	0	?	S	07:18	0:00	[ksoftirqd/1]
root	17	0.0	0.0	0	0	?	I	07:18	0:00	[kworker/1:0]
root	18	0.0	0.0	0	0	?	I<	07:18	0:00	[kworker/1:0H]
root	19	0.0	0.0	0	0	?	S	07:18	0:00	[cpuhp/2]
root	20	0.0	0.0	0	0	?	S	07:18	0:00	[watchdog/2]
root	21	0.0	0.0	0	0	?	S	07:18	0:00	[migration/2]
root	22	0.0	0.0	0	0	?	S	07:18	0:00	[ksoftirqd/2]

Figura 2 - Resultado do comando PS AUX.

Fonte: Elaborada pelo autor, 2019.

Entenda que um processo corresponde à representação de um programa, mas que não está sendo executado naquele momento. Ele possui estados predefinidos, conforme listamos a seguir. Clique e confira!

Execução

O processo está ativo, utilizando a CPU e outros recursos.

Pronto ou espera

O processo está temporariamente parado, permitindo que outro processo seja executado na sua frente.

Bloqueado

O processo está parado, aguardando a execução de algum evento para voltar ao estado de execução.

VOCÊ QUER VER?



Nos sistemas operacionais Linux, ao ser criado um processo, cria-se uma instância que permite a um *software* entrar em execução. Para outros sistemas, a criação de um processo se dá no momento em que o *software* é executado, e em muitos casos são denominadas “tarefas”, ou “*tasks*”. Por ser um sistema operacional multitarefa, o Linux permite que muitos dos recursos complexos e poderosos que lidam com uma grande quantidade de processos os executem de forma simultânea. Essa execução acontece quando diversos processos compartilham o uso do processador por meio de uma fila de processamento. No vídeo a seguir, você verá como trabalhar com processos em sistemas Linux. Confira: <https://youtu.be/a5m0aYqAuKo>.

Tenha em mente que as possibilidades do que podemos realizar por meio de processos são diversas. Conceitos de processos que geram outros processos-filhos são bem comuns, mas existe um controle para esses processos-pais e processos-filhos, que são feitos por meio de uma numeração denominada “PID”. Lembrando que, por meio da programação em Shell, também se cria ou se destrói processos, sendo possível até mesmo monitorar essas criações e destruições.

Existem diversos comandos que podem ser utilizados em scripts para uma finalidade específica, conforme o exemplo a seguir:

```
ps aux | sort -r -n --key=4 | awk ' $4 > 1.0 { print $4 " " $11 }'
```

Esse script, sendo executado em um terminal Shell, possibilita a visualização de todos os processos que estão consumindo um percentual “x” de memória RAM do computador. Vamos explicar cada parte do comando de forma mais detalhada. Continue acompanhando!

- **psaux**

mostra processos e informações de memória, CPU, dono, número do processo etc.

- **sort -r -n --key=4**

ordena, usando o quarto campo (memória) como chave e como campo numérico, do maior para o menor.

- **awk**

é uma linguagem de processamento de texto.

- **\$4 > 1.0**

faz filtrar apenas dados do quarto campo (memória) que sejam maiores que 1.0. Caso se coloque outros valores no lugar de 1.0, haverá uma mudança na exibição dos valores de uso da memória por meio do processo que se deseja analisar.

CASO

Os comandos do sistema operacional Linux são diversos, mas existem alguns mais conhecidos. Um deles é o FIND, pois sua utilização é bem comum, podendo ser considerado como um ponto



de partida para compreender as maneiras de se buscar um arquivo ou diretório em um sistema operacional Linux. Uma das formas mais simples de se utilizar o comando FIND é:

```
find . -name nome_do_arquivo.txt
```

Outro comando que pode ser usado para o mesmo fim é o comando LOCATE. Este comando é um *link* lógico que direciona para o comando MLOCATE, em que há a indexação de todo o sistema, contudo, o comando só pode realizar a indexação nos locais para os quais possui a devida permissão.

2.2.2 Outros comandos de gerenciamento

No terminal do Linux, há comandos que são responsáveis pelo gerenciamento de pacotes. Estes, por sua vez, em sua maioria são scripts que contêm regras para a instalação de pacotes no sistema operacional. Seu formato é do tipo *.tar.gz, também conhecidos como “tarball”, os quais frequentemente precisam ser compilados para poderem ser executados, ou seja, realizarem a instalação.

O primeiro passo para a utilização desse recurso é o processo de desempacotamento, feito por meio do comando “**tarxvfz programa.tar.gz**”. Esse comando é formado por vários comandos que, unidos, podem executar uma tarefa de descompactar um arquivo de nome “programa”. O comando responsável em fazer o processo de descompactação, contudo, é o TAR, sendo que o comando XVFZ corresponde apenas a opções de tarefas que o TAR irá realizar no processo de descompactação. Confira a seguir:

X abre o pacote;

V visualiza o tratamento dado;

F informa que será passado um arquivo;

Z informa que o pacote está compactado pela ferramenta gzip.

Todos os comandos do sistema Linux seguem uma sintaxe para que sua escrita esteja correta. Muitos dos comandos contêm dois hifens seguidos, e isso determina que logo em seguida deve ser colocado um ou mais caracteres para representar as opções do comando. Comandos que tenham colchetes servem para representar que há uma opção não obrigatória, o que significa que podem ser utilizados isoladamente ou com suas opções de configurações.

Também existe o uso de chaves para configurar formatos de ordenação, por exemplo. Da mesma forma, a barra vertical serve para que sejam buscadas opções excludentes entre si, fazendo com que os critérios do comando busquem o valor referenciado. Saiba que buscar formas de executar comandos intercalados do Linux, juntamente com os caracteres que possam agilizar ou otimizar o processo de execução do comando, é fundamental.

VAMOS PRATICAR?



Diante dessa explicação sobre alguns comandos de busca, verifique na Internet outras formas de implementação do comando LOCATE:

- “findutils”
- “slocate”
- “mlocate”

Verifique as bases de buscas que são criadas para cada um desses comandos.

2.3 Introdução ao terminal Linux e comandos de navegação em diretórios

A utilização de terminal é algo que vem dos primeiros computadores, os quais não disponibilizavam de interface gráfica para execução de comandos. Mesmo com a evolução das interfaces gráficas, o uso do terminal para realizar a navegação ainda é recorrente devido à velocidade em executar comandos e eficiência.

2.3.1 Console do sistema Linux - Terminal

Popularmente conhecido como “console”, o terminal do Linux é muito utilizado. Ele possibilita que o acesso seja feito por meio exclusivo do teclado, e que o usuário possa realizar a virtualização de terminais para acesso por meio de múltiplos usuários. Ele pode ser considerado uma segunda seção de trabalho a ser utilizada e que é totalmente independente das demais seções que estão ativas. Além disso, seu acesso pode ser feito local ou remotamente, via TELNET, RSH, RLOGIN, entre outros.

Clique nos itens e confira!

Mesmo assim, existem algumas diferenças que podem ser sutis entre o console e terminal, mas de necessário esclarecimento para que não haja uma interpretação dúbia entre esses dois conceitos. Todo e qualquer terminal é um programa dentro do qual é acionado um Shell para execução de comandos.

Antigamente, esses recursos eram imprescindíveis para que, por meio de um teclado e do monitor, pudéssemos usar o computador. Já o console é considerado um tipo de terminal que se utiliza de uma porta serial dedicada a um computador, que realiza uma comunicação direta por um nível mais baixo. Tudo isso dentro do sistema operacional.

Ao reproduzir a hierarquia do sistema por meio de camadas, podemos ver que a comunicação é feita por quatro níveis. O primeiro faz com que o usuário execute o Shell e, em seguida, envia as instruções aos serviços de baixo nível, os quais interagem com o kernel, e, finalmente, têm o devido acesso ao hardware do computador.

Diante dessas afirmativas, podemos ter a certeza de que o terminal do Linux é o instrumento que possibilita a utilização dos componentes do computador. Por meio dos comandos, temos amplo acesso aos recursos avançados do sistema. Tenha em mente que esse acesso pode ser ilimitado, dependendo do usuário que esteja utilizando.

VAMOS PRATICAR?



Em sistemas operacionais Linux para *desktop*, é necessário ter outro *software* para emular o terminal. Busque na Internet quais as possibilidades de *softwares* que podem realizar essa função.

2.3.2 Comandos de busca em Shell

O uso do terminal é de suma importância para quem deseja trabalhar com Shell Script. O terminal permite que operações complexas sejam feitas de forma mais ágil, devido ao fato de não demandar o uso de bibliotecas gráficas para realizar toda e qualquer tarefa. Temos, como exemplo, a velocidade com que se pode copiar um ou mais arquivos de um diretório local para um diretório externo em rede ou para um dispositivo externo.

VOCÊ SABIA?



Uma dica para ampliar seus conhecimentos em Shell é conhecer a diversidade de formas para acessar um terminal Shell, tanto no Linux quanto no Windows. Para ampliar seus conhecimentos, recomendamos a leitura do manual: https://linux.ime.usp.br/~lucasmmg/livecd/documentacao/documentos/terminal/terminal_basico.html.

O comando WHICH é utilizado na busca de comandos presentes no “*path*” do sistema do usuário corrente. Como a busca é feita dentro do *path*, ele mostrará os arquivos executáveis que se enquadram dentro dos parâmetros pré-determinados. Já o comando WHEREIS também faz a busca de arquivos executáveis, manuais, código-fonte e até mesmo de manuais que estejam agregados aos parâmetros das buscas. O mais interessante é que esse comando, em diversas buscas, traz mais informações ou conteúdos que o comando WHICH.

Uma forma de realizar buscas é por meio do comando FIND, que realiza uma busca por arquivos e diretórios dentro do sistema, baseado no que se deseja encontrar e nas configurações de pontos iniciais e finais da busca, conforme descritivo a seguir:

finddiretório_busca [opções][arquivo/diretório]

A opção denominada “diretório_busca” é o ponto inicial de busca. Ela também abrange os subdiretórios presentes dentro do mesmo diretório, sendo que as opções são divididas da seguinte forma:

- “*name*” (expressão) - realiza a busca pela expressão indicada (arquivo/diretório);
- “*size*” (tamanho) - realiza a busca pelo tamanho indicado. Caso a busca exceda o tamanho estipulado, pode-se usar “+” ou “-” antes de (tamanho);
- “*type*” (tipo) - realiza a busca pelo tipo de arquivo, mas seguindo quesitos baseados nas letras b – bloco; c – caractere; d – diretório; p – pipe; l – *link* simbólico; e s – socket;
- “*mindepth*” (num) - não faz a busca abaixo de subdiretórios com número de níveis;
- “*maxdepth*” (num) - faz a busca até em subdiretórios do diretório de busca.

Outra forma de realizar buscas é por meio do comando LOCATE/UPDATEDB. Esse comando realiza uma busca bem semelhante ao FIND, porém, sua busca é feita em bancos de dados que possam estar em um servidor local ou remoto. Sua sintaxe necessita apenas que seja colocada no terminal a palavra “*locate*” e, em seguida, o que se deseja procurar. Entretanto, antes deve ser construído o banco de dados por meio do comando UPDATEDB.

VAMOS PRATICAR?



Procure acessar o terminal de um sistema Linux e aplicar os comandos de busca para encontrar a pasta “documentos”. Faça uma pesquisa na Internet sobre como realizar buscas e procure aplicar a busca de outros diretórios e arquivos dentro do seu terminal.

2.4 Comandos de manipulação de arquivos e gestão de diretórios

A melhor compreensão de como utilizar o terminal vem por meio da manipulação de arquivos e diretórios do sistema Linux. Existem diversos comandos que permitem realizar tarefas de forma semelhante ao usuário que trabalha em modo gráfico. Todos esses comandos estão inclusos também quando se executam tarefas no modo gráfico.

2.4.1 Comandos para trabalhar arquivos e diretórios

Os sistemas Linux permitem realizar toda e qualquer manipulação de arquivos por meio do Shell. Muitos de seus comandos estão embutidos nos botões e ícones que são clicados com o *mouse*, mas isso não significa que ele depende exclusivamente de uma interface gráfica. Conforme você pode visualizar a seguir, a imagem representa os principais comandos de manipulação de diretórios:

- **cd** - navegar entre os diretórios
- **ls** - listar os arquivos
- **mkdir** - criar um diretório
- **rmdir** - remover um diretório vazio
- **cp** - copiar arquivos ou diretórios
- **rm** - deletar arquivos e diretórios
- **ln** - linkar arquivos
- **cat** - exibir o conteúdo de um arquivo ou direcionar para outro
- **file** - indica o tipo de arquivo

Figura 3 - Lista de comandos para manipular arquivos.

Fonte: Elaborada pelo autor, 2019.

O comando **CD** é responsável por realizar uma mudança de forma satisfatória do local onde está o usuário dentro da árvore de diretórios do sistema. Saiba que as abreviações são muito úteis para uma navegação rápida e mais eficiente. Essas abreviações são:

Abreviação	Significados
. (ponto)	Diretório atual
.. (dois pontos)	Diretório anterior
~(til)	Diretório HOME do usuário
/ (barra)	Diretório Raiz
- (hífen)	Último diretório

Quadro 1 - Abreviações do comando CD.

Fonte: Elaborada pelo autor, 2019.

Existem outros comandos para manipulação dos diretórios no sistema Linux. O comando MKDIR, por exemplo, permite a criação de um diretório e também permite ao usuário nomeá-lo, seguindo as mesmas regras que gerem o sistema operacional em modo visual.

Da mesma forma como há um comando para a criação do diretório, há o comando para remover ou apagar um diretório. Esse comando é o RMDIR, que é usado para apagar diretórios vazios. Caso deseje apagar um diretório independentemente do que esteja dentro dele, é necessário usar o comando RM.

Uma das funções mais intrigantes do sistema Linux é a sua capacidade de realizar cópias de arquivos de forma rápida e eficiente, independentemente da quantidade de informações que eles contenham. O comando que realiza essa proeza de forma tão fantástica é o CP. Ele tem várias configurações, que nos permitem fazer cópias de forma mais veloz e precisa. A seguir, observe as configurações do comando CP.

Parâmetro	Significado
-i	Modo interativo
-v	Mostra o que está sendo copiado
-R	Copia recursivamente (diretórios e subdiretórios)

Quadro 2 - Abreviações do comando CP.

Fonte: Elaborada pelo autor, 2019.

O ponto mais intrigante desse comando é a opção do modo interativo, que possibilita a um arquivo transcrever sobre outro, embora antes o sistema pergunte se o usuário realmente deseja executar aquela tarefa. Vale ressaltar que essa execução não necessariamente precisa que os arquivos tenham o mesmo nome.

A união de conteúdo de arquivos também é possível no Shell por meio de um comando denominado "CAT". Esse comando tanto permite que seja feita a exibição do conteúdo de um arquivo quanto o envio de seu conteúdo para outro. A funcionalidade mais interessante, como mencionamos, é a possibilidade de enviar o conteúdo de um arquivo para outro, realizando a junção desses arquivos. Ele também permite executar um arquivo de som, enviando-o para o respectivo *software* responsável por executá-lo.

VAMOS PRATICAR?



A diversidade de comandos citados permite que muitas tarefas sejam executadas por meio deles. Busque na Internet as formas de aplicar vários comandos de forma sequencial, evitando, assim, a necessidade de aplicar um comando por vez.

Síntese

Chegamos ao final desta unidade. Aqui, abordamos comandos que serão utilizados para uma melhor administração de servidores Linux, e que, por meio do Shell Script, podem automatizar diversos recursos. Neste momento, você já sabe como é o funcionamento dos pacotes e manipulação de arquivos de sistemas Linux.

Nesta unidade, você teve a oportunidade de:

- compreender as permissões de diretórios;
- saber como é o funcionamento dos pacotes em servidores Linux;
- entender como trabalhar com arquivos e diretórios;
- entender quais configurações podem ser utilizadas em comandos;
- conhecer a pesquisa de arquivos em diretórios;
- saber como realizar a cópia de arquivos de um local para outro;
- entender como fazer a leitura do conteúdo de um arquivo;
- entender como se descompacta arquivos no Linux.

Bibliografia

CANALTECH. Estudo mostra que 83% das empresas executam Linux em seus servidores, 18 set. 2013. Disponível em: <https://canaltech.com.br/linux/Estudo-mostra-que-83-das-empresas-executam-Linux-em-seus-servidores/>. Acesso em: 30/09/2019.

KURTPHPR. **Aula 17 Linux** - Controle de Processos, background, foreground, ps, top e kill, 17 mar. 2010. Disponível em: <https://youtu.be/a5mOaYqAuKo>. Acesso em: 30/09/2019.

MONQUEIRO, J. C. B. Linux: Gerenciamento de usuários, grupos e permissões. **Hardware.com.br**, 18 jun. 2008. Disponível em: <https://www.hardware.com.br/tutoriais/usuarios-grupos-permissoes/>. Acesso em: 30/09/2019.

NEGUS, C. **Linux** – a Bíblia. Rio de Janeiro: Altas Books, 2014.

NEMETH, E. et al. **Manual Completo do Linux**. São Paulo: Pearson Prentice Hall, 2007.

ROMERO, D. **Começando com Linux** - comandos, serviços e administração. São Paulo: Casa do Código, 2014.

SHIELDS, I. Gerenciamento de pacote RPM e YUM, 13 dez. 2010. **IBM** – Série Aprenda Linux, 101. Disponível em: <https://www.ibm.com/developerworks/br/linux/library/l-lpic1-v3-102-5/index.html>. Acesso em: 30/09/2019.

SILVA, G. N. **Como usar o APT**. Capítulo 5 - Obtendo informações sobre os pacotes, ago. 2015. Disponível em: <https://www.debian.org/doc/manuals/apt-howto/ch-search.pt-br.html>. Acesso em: 30/09/2019.

SOARES, W.; FERNANDES, G. **Linux Fundamentos**. São Paulo: Érica, 2010.

STEVENS, W. R. et al. **Programação de rede Unix**. 3. ed. Porto Alegre: Bookman, 2005.

TANENBAUM, A. S.; WOODHULL, A. S. **Sistemas Operacionais Modernos**. 3. ed. Porto Alegre: Bookman, 2008.

TERMINAL básico de rede Linux. Disponível em: https://linux.ime.usp.br/~lucasmmg/livecd/documentacao/documentos/terminal/Terminal_basico.html. Acesso em: 02/10/2019.

VENDRAMETO. I. **Utilizando Shell Script** - Parte 1, 14 maio 2019. Disponível em: <https://inside.contabilizei.com.br/utilizando-shell-script-parte-1-2dd9dcf8e9dd>. Acesso em: 30/09/2019.