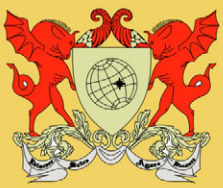
Material ampliado e adaptado do Guia Foca Linux

Curso Básico de LINUX

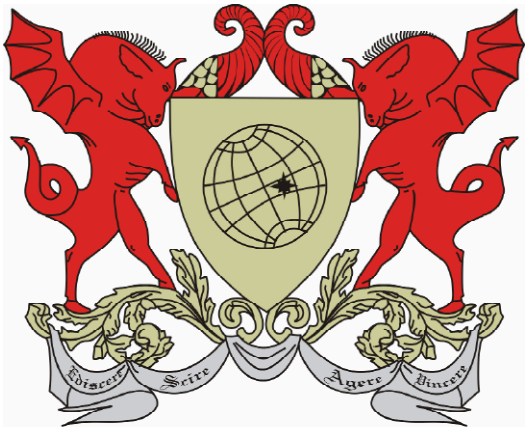
Módulo II



Universidade Federal de Viçosa

cead

Coordenadoria de Educação Aberta e à Distância



Universidade Federal de Viçosa

Nilda de Fátima Ferreira Soares - Reitora

Demétrius David da Silva - Vice-Reitor

CEAD - Coodenadoria de Educação Aberta e a Distância

Frederico Vieira Passos - Diretor

VAREJÃO-JR, C.G; COSTA,M.H. - Curso de Linux - Módulo I. Viçosa, 2010.

Layout: José Timóteo Júnior

Edição de imagens e capa: Diogo Rodrigues

Editoração Eletrônica: Hamilton Henrique Teixeira Reis, Rômulo Siqueira Santos

Revisão: Diogo Rodrigues

Revisão Final: João Batista Mota

CEAD - Prédio CEE, Avenida PH Rolfs s/n
Campus Universitário, 36570-000, Viçosa/MG
Telefone: (31) 3899 2858 | Fax: (31) 3899 3352

Sumário

Curso Básico de Linux
Módulo II

Capítulo 01 - Comandos diversos

Veremos alguns comandos de uso diversos no sistema ao longo do capítulo.

1.1 clear

Limpa a tela e posiciona o cursor no canto superior esquerdo do vídeo.

clear

1.2 date

Permite ver ou modificar a data e a hora do sistema. Você precisa estar como usuário root para modificar data e hora. Muitos programas do sistema, arquivos de registro (log) e tarefas agendadas funcionam com base na data e hora fornecidas pelo sistema. Sendo assim, esteja consciente das modificações que data e hora podem trazer a esses programas (principalmente se tratando de uma rede com muitos usuários).

date MesDiaHoraMinuto[AnoSegundos]

Onde:

MesDiaHoraMinuto[AnoSegundos]

São respectivamente os números do mês, dia, hora e minutos sem espaços.

Opcionalmente você pode especificar o ano (com 2 ou 4 dígitos) e os segundos.

+[FORMATO]

- Define o formato da listagem que será usada pelo comando date. Os seguintes formatos são os mais usados:%d - Dia do mês (00-31).
  - %m - Mês do ano (00-12).
  - %y - Ano (dois dígitos).
  - %Y - Ano (quatro dígitos).
  - %H - Hora (00-24).
  - %I - Hora (00-12).
  - %M - Minuto (00-59).
  - %j - Dia do ano (1-366).
  - %p - AM/PM (útil se utilizado com %d).
  - %r - Formato de 12 horas completo (hh:mm:ss AM/PM).
  - %T - Formato de 24 horas completo (hh:mm:ss).
  - %w - Dia da semana (0-6).

Outros formatos podem ser obtidos através da página de manual do date.

Para maiores detalhes, veja a página de manual do comando date.

Para ver a data atual digite: date

Se quiser mudar a Data para 25/12 e a hora para 08:15 digite: date 12250815

Para mostrar somente a data no formato dia/mês/ano: date +%d/%m/%Y

1.3 df

Mostra o espaço livre/ocupado de cada partição.

df [opções]

Onde:

opções

-a

Inclui sistemas de arquivos com 0 blocos.

-h, --human-readable

Mostra o espaço livre/ocupado em MB, KB, GB ao invés de blocos.

-H

Idêntico a -h mas usa 1000 ao invés de 1024 como unidade de cálculo.

-k

Lista em Kbytes.

-l

Somente lista sistema de arquivos locais.

-m

Lista em Mbytes (equivalente a --block-size=1048576).

--sync

Executa o sync antes de mostrar os dados.

-T

Lista o tipo de sistema de arquivos de cada partição

-t tipo

Lista somente sistema de arquivos do tipo tipo.

-x tipo

Não lista sistemas de arquivos do tipo tipo.

Exemplos: df, df -h, df -t vfat.

1.4 ln

Cria links para arquivos e diretórios no sistema. O link é um mecanismo que faz referência a outro arquivo ou diretório em outra localização. O link em sistemas GNU/Linux faz referência real ao arquivo/diretório, podendo ser feita cópia do link (será copiado o arquivo alvo), entrar no diretório (caso o link faça referência a um diretório), etc.

ln [opções] [origem] [link]

Onde:

origem

Diretório ou arquivo de onde será feito o link.

link

Nome do link que será criado.

opções

-s

Cria um link simbólico. Usado para criar ligações com o arquivo/diretório de destino.

-v

Mostra o nome de cada arquivo antes de fazer o link.

-d

Cria um hard link para diretórios. Somente o root pode usar esta opção.

- Existem dois tipos de links: simbólicos e hardlinks. O **link simbólico** cria um arquivo especial no disco (do tipo link), que tem como conteúdo o caminho para chegar até o arquivo alvo (isso pode ser verificado pelo tamanho do arquivo do link). Use a opção -s para criar links simbólicos.

- O **hardlink** faz referência ao mesmo inodo do arquivo original. Dessa forma, ele será perfeitamente idêntico, inclusive nas permissões de acesso ao arquivo original.

Ao contrário dos links simbólicos, não é possível fazer um hardlink

para um diretório ou fazer referência a arquivos que estejam em partições diferentes.

Observações:

- Se for usado o comando rm com um link, somente o link será removido.
- Se for usado o comando cp com um link, o arquivo original será copiado ao invés do link.
- Se for usado o comando mv com um link, a modificação será feita no link.
- Se for usado um comando de visualização (como o cat), o arquivo original será visualizado.

Exemplos:

- ln -s /dev/ttyS1 /dev/modem - Cria o link /dev/modem para o arquivo /dev/ttyS1.
- ln -s /tmp ~/tmp - Cria um link ~/tmp para o diretório /tmp.

1.5 du

Mostra o espaço ocupado por arquivos e sub-diretórios do diretório atual.

du [opções]

Onde:

opções

-a, --all

Mostra o espaço ocupado por todos os arquivos.

-b, --bytes

Mostra o espaço ocupado em bytes.

-c, --total

Faz uma totalização de todo espaço listado.

-D

Não conta links simbólicos.

-h, --human

Mostra o espaço ocupado em formato legível por humanos (Kb, Mb) ao invés de usar blocos.

-H

Como o anterior mas usa 1000 e não 1024 como unidade de cálculo.

-k

Mostra o espaço ocupado em Kbytes.

-m

Mostra o espaço ocupado em Mbytes.

-S, --separate-dirs

Não calcula o espaço ocupado por sub-diretórios.

-x

Não faz a contagem de diretórios em sistemas de arquivos diferentes do atual.

Exemplo: du -h, du -hc.

1.6 find

Procura por arquivos/diretórios no disco. find pode procurar arquivos através de sua data de modificação, tamanho, etc através do uso de opções. find, ao contrário de outros programas, usa opções longas através de um "-".

find [diretório] [opções/expressão]

Onde:

diretório

Inicia a procura neste diretório, percorrendo seus sub-diretórios.

opções/expressão

-name [expressão]

Procura pelo nome [expressão] nos nomes de arquivos e diretórios processados.

-depth

Processa os sub-diretórios primeiro antes de processar os arquivos do diretório principal.

-maxdepth [num]

Faz a procura até [num] sub-diretórios dentro do diretório que está sendo pesquisado.

-mindepth [num]

Não faz nenhuma procura em diretórios menores que [num] níveis.

-mount, -xdev

Não faz a pesquisa em sistemas de arquivos diferentes daquele de onde o comando find foi executado.

-amin [num]

Procura por arquivos que foram acessados [num] minutos atrás. Caso for antecedido por "-", procura por arquivos que foram acessados entre [num] minutos atrás e agora.

-atime [num]

Procura por arquivos que foram acessados [num] dias atrás. Caso for antecedido por "-", procura por arquivos que foram acessados entre [num] dias atrás e a data atual.

-gid [num]

Procura por arquivos que possuam a identificação numérica do grupo igual a [num].

-group [nome]

Procura por arquivos que possuam a identificação de nome do grupo igual a [nome].

-uid [num]

Procura por arquivos que possuam a identificação numérica do usuário igual a [num].

-user [nome]

Procura por arquivos que possuam a identificação de nome do usuário igual a [nome].

-inum [num]

Procura por arquivos que estão localizados no inodo [num].

-links [num]

Procura por arquivos que possuem [num] links como referência.

-mmin [num]

Procura por arquivos que tiveram seu conteúdo modificado há [num] minutos. Caso for antecedido por "-", procura por arquivos que tiveram seu conteúdo modificado entre [num] minutos atrás e agora.

-mtime [num]

Procura por arquivos que tiveram seu conteúdo modificado há [num] dias. Caso for antecedido por "-", procura por arquivos que tiveram seu conteúdo modificado entre [num] dias atrás e agora.

-ctime [num]

Procura por arquivos que teve seu status modificado há [num] dias. Caso for antecedido por "-", procura por arquivos que tiveram seu conteúdo modificado entre [num] dias atrás e agora.

-nouser

Procura por arquivos que não correspondam à identificação do usuário atual.

-nogroup

Procura por arquivos que não correspondam à identificação do grupo do usuário atual.

-perm [modo]

Procura por arquivos que possuam os modos de permissão [modo]. O [modo] de permissão pode ser numérico (octal) ou literal.

-used [num]

Procura por arquivos que foram acessados [num] dias depois de ter seu status modificado.

-size [num]

- Procura por arquivos que tiverem o tamanho [num]. [num] pode ser antecedido de "+" ou "-" para especificar um arquivo maior ou menor que [num]. A opção -size pode ser

seguida de:b - Especifica o tamanho em blocos de 512 bytes. É o padrão caso [num] não seja acompanhado de nenhuma letra.

- c - Especifica o tamanho em bytes.
- k - Especifica o tamanho em Kbytes.

-type [tipo]

Procura por arquivos do [tipo] especificado. Os seguintes tipos são aceitos:

- b - bloco
- c - caracter
- d - diretório
- p - pipe
- f - arquivo regular
- l - link simbólico
- s - soquete

A maior parte dos argumentos numéricos podem ser precedidos por “+” ou “-”. Para detalhes sobre outras opções e argumentos, consulte a página de manual.

Exemplo:

- find / -name grep - Procura no diretório raíz e sub-diretórios um arquivo/diretório chamado grep.
- find / -name grep -maxdepth 3 - Procura no diretório raíz e sub-diretórios até o 3o. nível, um arquivo/diretório chamado grep.
- find . -size +1000k - Procura no diretório atual e sub-diretórios um arquivo com tamanho maior que 1000 kbytes (1Mbyte).
- find / -mmin 10 - Procura no diretório raíz e sub-diretórios um arquivo que foi modificado há 10 minutos atrás.
- find / -links 4 - Procura no diretório raíz e sub-diretórios, todos os arquivos que possuem 4 links como referência.

1.7 free

Mostra detalhes sobre a utilização da memória RAM do sistema.

free [opções]

Onde:

opções

-b

Mostra o resultado em bytes.

-k

Mostra o resultado em Kbytes.

-m

Mostra o resultado em Mbytes.

-o

Oculta a linha de buffers.

-t

Mostra uma linha contendo o total.

-s [num]

Mostra a utilização da memória a cada [num] segundos.

O free é uma interface ao arquivo /proc/meminfo.

1.8 grep

Procura por um texto dentro de um ou mais arquivos ou no dispositivo de entrada padrão.

grep [expressão] [arquivo] [opções]

Onde:

expressão

palavra ou frase que será procurada no texto. Se tiver mais de 2 palavras você deve identifica-la com aspas “” caso contrário o grep assumirá que a segunda palavra é o arquivo!

arquivo

Arquivo onde será feita a procura.

opções

-A [número]

Mostra as [número] linhas após a linha encontrada pelo grep.

-B [número]

Mostra as [número] linhas antes da linha encontrada pelo grep.

-f [arquivo]

Especifica que o texto a ser localizado está no arquivo [arquivo].

-h, --no-filename

Não mostra os nomes dos arquivos durante a procura.

-i, --ignore-case

Ignora diferença entre maiúsculas e minúsculas no texto procurado e arquivo.

-n, --line-number

Mostra o nome de cada linha encontrada pelo grep.

-U, --binary

Trata o arquivo que será procurado como binário.

-v

Mostra as linhas onde não aparece a string

Se não for especificado o nome de um arquivo ou se for usado um hífen “-”, grep procurará a string no dispositivo de entrada padrão.

O grep faz sua pesquisa em arquivos texto. Use o comando zgrep para pesquisar diretamente em arquivos compactados com gzip. Os comandos e opções são as mesmas.

Exemplos:

grep “capitulo” texto.txt; ps ax | grep inetd; grep “capitulo” texto.txt -A 2 -B 2.

1.9 head

Mostra as linhas iniciais de um arquivo texto.

head [opções]

Onde:

opções

-c [numero]

Mostra o [numero] de bytes do início do arquivo.

-n [numero]

Mostra o [numero] de linhas do início do arquivo. Caso não for especificado, o head mostra as 10 primeiras linhas.

Exemplos: head teste.txt, head -n 20 teste.txt.

1.10 nl

Mostra o número de linhas junto com o conteúdo de um arquivo.

nl [*opções*] [*arquivo*]

Onde:

opções

-f [opc]

Faz a filtragem de saída de acordo com [opc]:

a

Numera todas as linhas.

t

Não numera linhas vazias.

n

Numera linhas vazias.

texto

Numera somente linhas que contenham o [texto].

-v [num]

Número inicial (o padrão é 1).

-i [num]

Número de linhas adicionadas a cada linha do arquivo (o padrão é 1).

Exemplos: nl /etc/passwd, nl -i 2 /etc/passwd.

1.11 more

Permite fazer a paginação de arquivos ou da entrada padrão. O comando *more* pode ser usado como comando para leitura de arquivos que ocupem mais de uma tela. Quando a tela fica toda cheia, o *more* efetua uma pausa e permite que você pressione <ENTER> ou <ESPAÇO> para continuar avançando no arquivo sendo visualizado. Para sair do *more* pressione q.

more [arquivo]

Onde: *arquivo* é o arquivo que será paginado.

Para visualizar diretamente arquivos texto compactados pelo gzip .gz, use o comando zmore.

Exemplos: more /etc/passwd, cat /etc/passwd | more.

1.12 less

Permite fazer a paginação de arquivos ou da entrada padrão. O comando less pode ser usado como comando para leitura de arquivos que ocupem mais de uma tela. Quando toda a tela é ocupada, o less efetua uma pausa (semelhante ao more) e permite que você pressione <SETA PARA CIMA>/<SETA PARA BAIXO> ou <PAGE UP>/<PAGE DOWN> para fazer o rolamento da página. Para sair do less, pressione q.

less [arquivo]

Onde: *arquivo* é o arquivo que será paginado.

Para visualizar diretamente arquivos texto compactados pelo utilitário gzip (arquivos .gz), use o comando zless.

Exemplos: less /etc/passwd, cat /etc/passwd | less

1.13 sort

Organiza as linhas de um arquivo texto ou da entrada padrão. A organização é feita por linhas.

sort [opções] [arquivo]

Onde:

arquivo

É o nome do arquivo que será organizado. Caso não for especificado, será usado o dispositivo de entrada padrão (normalmente o teclado ou um “|”).

opções

-b

Ignora linhas em branco.

-d

Somente usa letras, dígitos e espaços durante a organização.

-f

Ignora a diferença entre maiúsculas e minúsculas.

-r

Inverte o resultado da comparação.

-n

Caso estiver organizando um campo que contém números, eles serão organizados na ordem aritmética. Por exemplo, se você tiver um arquivo com os números

100

10

50

Usando a opção -n, o arquivo será organizado desta maneira:

10

50

100

Caso esta opção **não** for usada com o sort, ele organizará como uma listagem alfabética (que vai de a até z e de 0 a 9)

10

100

50

-c

Verifica se o arquivo já está organizado. Caso não estiver, retorna a mensagem “disorder on *arquivo*”.

-o arquivo

Grava a saída do comando sort no arquivo.

-m arquivo1 arquivo2

Combina o conteúdo de *arquivo1* e *arquivo2,* gerando um único arquivo. Os dois arquivos precisam estar ordenados antes de se utilizar esta opção.

-i

Ignora os caracteres fora da faixa octal ASCII 040-0176 durante a organização.

-t caracter

Usa caracter como delimitador durante a organização de linhas. Por padrão é usado um espaço em branco como delimitador de caracteres.

+num1 -num2

Especifica qual o campo dentro na linha que será usado na organização. O(s) campo(s) usado(s) para organização estará(ão) entre +num1 e +num2. O delimitador padrão utilizado é um espaço em branco (use a opção -t para especificar outro). A contagem é iniciada em “0”. Caso não for especificada, a organização é feita no primeiro campo. Caso -num2 não seja especificado, a organização será feita usando a coluna +num1 até o fim da linha.

-k num1, num2

Essa é uma alternativa ao método acima para especificar as chaves de organização. O uso é idêntico, mas o delimitador é iniciado em “1”.

- Abaixo, exemplos de uso do comando sort texto.txt - Organiza o arquivo texto.txt em ordem crescente.
- sort texto.txt -r - Organiza o conteúdo do arquivo texto.txt em ordem decrescente.
- cat texto.txt|sort - Faz a mesma coisa que o primeiro exemplo, só que neste caso a saída do comando cat é redirecionado para a entrada padrão do comando sort.
- sort -f texto.txt - Ignora diferenças entre letras maiúsculas e minúsculas durante a organização.
- sort +1 -3 texto.txt - Organiza o arquivo texto.txt usando como referência a segunda até a quarta palavra (segundo ao quarto campo) que constam naquela linha.
- sort -t : +2 -3 passwd - Organiza o arquivo passwd usando como referência a terceira até a quarta palavra (terceiro ao quarto campo). Note que a opção -t especifica o caracter “:”, ao invés do espaço, como delimitador de campos. Nesse caso, o que estiver após “:” será considerado o próximo campo.

1.14 tail

Mostra as 10 linhas finais de um arquivo texto.

tail [opções] arquivo

Onde:

opções

-c [numero]

Mostra o [numero] de bytes do final do arquivo.

-n [numero]

Mostra o [numero] de linhas do final do arquivo (o padrão é 10 linhas).

-f

Continua indefinidamente tentando ler caracteres ao final do arquivo, assumindo que o arquivo está crescendo. Útil para monitorar arquivos de log, por exemplo.

Exemplos: tail teste.txt, tail -n 20 teste.txt.

1.15 time

Mede o tempo gasto para executar um processo (programa).

time [comando]

Onde: *comando* é o comando/programa para o qual se deseja medir o tempo gasto para ser concluído.

Exemplo: time ls, time find / -name crontab.

1.16 touch

Muda a data e hora em que um arquivo foi criado. Também pode ser usado para criar arquivos vazios. Caso o touch seja usado com arquivos que não existam, por padrão ele criará estes arquivos.

touch [opções] [arquivos]

Onde:

arquivos

Arquivos que terão sua data e/ou hora modificados.

opções

-t MMDDhhmm[ANO.segundos]

Usa Meses (MM), Dias (DD), Horas (hh), minutos (mm) e opcionalmente o ANO e segundos para modificação do(s) arquivo(s) ao invés da data e hora atuais.

-a, --time=atime

Faz o touch mudar somente a data e hora do acesso ao arquivo.

-c, --no-create

Não cria arquivos vazios, caso os *arquivos* não existam.

-m, --time=mtime

Faz o touch mudar somente a data e hora da modificação.

-r [arquivo]

Usa as horas no [arquivo] como referência ao invés da hora atual.

- Exemplostouch teste - Cria o arquivo teste caso ele não existir.
- touch -t 10011230 teste - Altera da data e hora do arquivo para 01/10 e 12:30.
- touch -t 120112301999.30 teste - Altera da data, hora ano, e segundos do arquivo para 01/12/1999 e 12:30:30.
- touch -t 12011200 * - Altera a data e a hora de todos os arquivos do diretório atual para 01/12 e 12:00.

1.17 uptime

Mostra o tempo de execução do sistema desde que o computador foi ligado.

uptime

1.18 dmesg

Mostra as mensagens de inicialização do kernel. São mostradas as mensagens da última inicialização do sistema.

dmesg|less

1.19 mesg

Permite ou não o recebimento de requisições de talk de outros usuários.

mesg [y/n]

Onde: *y* permite que você receba “talks” de outros usuários.

Digite mesg para saber se você pode ou não receber “talks” de outros usuários. Caso a resposta seja “n” você poderá enviar um talk para alguém mas o seu sistema se recusará em receber talks de outras pessoas.

É interessante colocar o comando mesg y em seu arquivo de inicialização .bash_profile para permitir o recebimento de “talks” toda vez que entrar no sistema.

Detalhes sobre como se comunicar com outros usuários serão vistos à frente.

1.20 echo

Mostra mensagens. Este comando é útil na construção de scripts para mostrar mensagens na tela para o usuário acompanhar sua execução.

echo [mensagem]

A opção -n pode ser usada para que não ocorra o salto de linha após a mensagem ser mostrada.

1.21 su

Permite ao usuário mudar sua identidade para outro usuário sem fazer o logout. Útil para executar um programa ou comando como root sem ter que abandonar a seção atual.

su [usuário]

Onde: *usuário* é o nome do usuário que deseja usar para acessar o sistema. Se não digitado, é assumido o usuário root.

Será pedida a senha do superusuário para autenticação. Digite exit quando desejar retornar a identificação de usuário anterior.

1.22 sync

Grava os dados do cache de disco na memória RAM para todos os discos rígidos e flexíveis do sistema. O cache é um mecanismo de aceleração, que permite o armazenamento na memória, ao invés de ser imediatamente gravado no disco. Quando o sistema estiver ocioso, o arquivo é gravado para o disco. O GNU/Linux procura utilizar toda memória RAM disponível para o cache de programas acelerando seu desempenho de leitura/gravação.

sync

O uso do sync é útil em disquetes quando gravamos um programa e precisamos que os dados sejam gravados imediatamente para retirar o disquete da unidade. Mas o método recomendado é especificar a opção sync durante a montagem da unidade de disquetes, como visto anteriormente.

1.23 uname

Retorna o nome e versão do kernel atual.

uname

1.24 reboot

Reinicia o computador.

1.25 shutdown

Desliga/reinicia o computador imediatamente ou após determinado tempo (programável) de forma segura. Todos os usuários do sistema são avisados que o computador será desligado. Esse comando somente pode ser executado pelo usuário root ou quando é usada a opção -a pelos usuários cadastrados no arquivo /etc/shutdown.allow que estejam logados no console virtual do sistema.

shutdown [opções] [hora] [mensagem]

Onde:

hora

Momento que o computador será desligado. Você pode usar HH:MM para definir a hora e minuto, MM para definir minutos, +SS para definir após quantos segundos, ou now para imediatamente (equivalente a +0).

O shutdown criará o arquivo /etc/nologin para não permitir que novos usuários façam login no sistema (com excessão do root). Esse arquivo é removido caso a execução do shutdown seja cancelada (opção -c) ou após o sistema ser reiniciado.

mensagem

Mensagem que será mostrada a todos os usuários, alertando sobre o reinício/desligamento do sistema.

opções

-h

Inicia o processo para desligamento do computador.

-r

Reinicia o sistema

-c

Cancela a execução do shutdown. Você pode acrescentar uma mensagem avisando aos usuários sobre o fato.

-a

Permite que os nomes de usuários contidos no arquivo /etc/shutdown.allow possam utilizar o shutdown para reinicializar/desligar o sistema. Deve ser colocado um nome de usuário por linha. O limite máximo de usuários neste arquivo é de 32.

Esse arquivo é útil quando o shutdown é usado para controlar o pressionamento das teclas <CTRL+ALT+DEL> no /etc/inittab.

-k

Simula o desligamento/reinício do sistema, enviando mensagem aos usuários.

-f

Não executa a checagem do sistema de arquivos durante a inicialização do sistema. Esse processo é feito gravando-se um arquivo /fastboot, que é interpretado pelos scripts responsáveis pela execução do fsck durante a inicialização do sistema.

-F

Força a checagem do sistema de arquivos durante a inicialização. É gravado um arquivo chamado /forcefsck, que é interpretado pelos scripts responsáveis pela execução do fsck durante a inicialização do sistema.

-n

Faz com que o shutdown ignore a execução do init, fechando todos os processos.

-t [num]

Faz com que o shutdown envie um sinal de término aos processos e aguarde [num] segundos antes de enviar o sinal KILL.

O shutdown envia uma mensagem a todos os usuários do sistema, alertando sobre o desligamento durante os 15 minutos restantes, e assim permite que finalizem suas tarefas. Depois disso, o shutdown muda o nível de execução, através do comando init, para 0 (desligamento), 1 (modo monousuário), 6 (reinicialização). É recomendado utilizar o símbolo “&” no final da linha de comando, para que o shutdown seja executado em segundo plano.

Quando restarem apenas 5 minutos para o reinício/desligamento do sistema, o programa login será desativado, impedindo a entrada de novos usuários no sistema.

O programa shutdown pode ser chamado pelo init através do pressionamento da combinação das teclas de reinicialização <CTRL+ALT+DEL>, alterando-se o arquivo /etc/inittab. Isso permite que somente os usuários autorizados (ou o root) possam reiniciar o sistema.

- Exshutdown -h now - Desligar o computador imediatamente.
- shutdown -r now - Reinicia o computador imediatamente.
- shutdown 19:00 “A manutenção do servidor será iniciada às 19:00” - Faz o computador entrar em modo monousuário (init 1) às 19:00 enviando a mensagem “*A manutenção do servidor será iniciada às 19:00”* a todos os usuários conectados ao sistema.
- shutdown -r 15:00 “O sistema será reiniciado às 15:00 horas” - Faz o computador ser reiniciado (init 6) às 15:00 horas enviando a mensagem *O sistema será reiniciado às 15:00*

horas a todos os usuários conectados ao sistema.

- shutdown -r 20 - Faz o sistema ser reiniciado após 20 minutos.
- shutdown -c - Cancela a execução do shutdown.
- shutdown -t 30 -r 20 - Reinicia o sistema após 20 minutos, espera 30 segundos após o sinal de término para enviar o sinal KILL a todos os programas abertos.

1.26 wc

Conta o número de palavras, bytes e linhas em um arquivo ou entrada padrão. Se as opções forem omitidas, o wc mostra a quantidade de linhas, palavras, e bytes.

wc [opções] [arquivo]

Onde:

arquivo

Arquivo que será verificado pelo comando wc.

opções

-c, --bytes

Mostra os bytes do arquivo.

-w, --words

Mostra a quantidade de palavras do arquivo.

-l, --lines

Mostra a quantidade de linhas do arquivo.

A ordem da listagem dos parâmetros é única, e modificando a posição das opções não modifica a ordem que os parâmetros são listados.

- Exemplo: wc /etc/passwd - Mostra a quantidade de linhas, palavras e letras (bytes) no arquivo /etc/passwd.
- wc -w /etc/passwd - Mostra a quantidade de palavras.
- wc -l /etc/passwd - Mostra a quantidade de linhas.
- wc -l -w /etc/passwd - Mostra a quantidade de linhas e palavras no arquivo /etc/passwd.

1.27 seq

Imprime uma seqüência de números começando em [primeiro] e terminando em [último], utilizando [incremento] para avançar.

seq [opções] [primeiro] [incremento] [último]

Onde:

primeiro

Número inicial da seqüência.

incremento

Número utilizado para avançar na seqüência.

último

Número final da seqüência.

opções

-f, --format=[formato]

Formato de saída dos números da seqüência. Utilize o estilo do printf para ponto flutuante (valor padrão: %g).

-s, --separator=[string]

Usa [string] para separar a seqüência de números (valor padrão: \n).

-w, --equal-width

Insere zeros na frente dos números mantendo a seqüência alinhada.

- Observações: Se [primeiro] ou [incremento] forem omitidos, o valor padrão 1 será utilizado.
- Os números recebidos são interpretados como números em ponto flutuante.
- [incremento] deve ser positivo se [primeiro] for menor do que o último, e negativo caso contrário.
- Quando utilizarmos a opção --format, o argumento deve ser exatamente %e, %f ou %g.

Exemplos:seq 0 2 10, seq -w 0 10, seq -f%f 0 10, seq -s", " 0 10

1.28 chattr

Modifica atributos de arquivos/diretórios. Não confunda atributos de arquivo com permissões de acesso. Os atributos são diferentes e definem outras características especiais para os arquivos/diretórios especificados.

chattr [opções] [atributos] [arquivos/diretórios]

Onde:

arquivos/diretórios

Arquivos/diretórios que terão os atributos modificados. Podem ser usados curingas

opções

-R

Modifica atributos em subdiretórios

-V

Mostra detalhes sobre a modificação de atributos.

atributos

- Os atributos de arquivos/diretórios podem ser especificados da seguinte maneira: + - Adiciona o atributo
- - - Remove o atributo
- = - Define o atributo exatamente como especificado

Os atributos são os seguintes:

- A - Não modifica a hora de acesso de arquivos. Poder aumentar consideravelmente a performance em Notebooks devido à diminuição de I/O no disco rígido. Quando especificada

em diretórios, faz com que todos os arquivos e subdiretórios residentes nele não tenham a hora de acesso modificada.

Esse atributo funciona apenas em kernels 2.2 e superiores.

- a - Append-Only - Arquivos com este atributo podem somente ser gravados em modos incrementais (o conteúdo pode ser adicionado somente ao final do arquivo). Eles não podem ser removidos, renomeados e novos links não poderão ser criados para estes arquivos.

Em diretórios faz com que os arquivos sejam apenas adicionados. Somente o root pode especificar ou retirar este atributo.

- c - Permite compactação nos arquivos especificados, de forma transparente para o usuário. Durante a leitura, o kernel retorna dados descompactados e, durante a gravação, os dados são compactados e gravados no disco.

Este atributo ainda não foi totalmente implementado no código atual do kernel.

- d - Este atributo não é usado pelo kernel, mas faz o programa dump evitar backup dos arquivos marcados com este atributo.

- i - Imutável - Arquivos imutáveis não podem ser modificados, os dados também não podem ser gravados para estes arquivos, não podem ser removidos nem renomeados. Até mesmo o usuário root não pode modificá-los.

Em diretórios, faz com que arquivos não possam ser adicionados ou apagados. Somente o usuário root pode especificar ou retirar esse atributo.

- s - O arquivo especificado é marcado como “apagamento seguro”. Quando o arquivo é apagado, seus blocos são zerados e gravados de volta no disco (eliminando qualquer possibilidade de recuperação).

- S - Faz a gravação imediatamente para o arquivo especificado. É como especificar a opção “sync” na montagem do sistema de arquivos ext2, mas afeta somente os arquivos especificados. Não tem efeito em diretórios.

- u - O arquivo especificado é marcado como recuperável. Quando o arquivo é apagado, seu conteúdo é salvo para permitir futura recuperação.

Este atributo ainda não foi implementado totalmente no código atual do kernel.

Os atributos de arquivos/diretórios são visualizados através do utilitário lsattr. Existem patches para os kernels da série 2.2 que adicionam o suporte experimental aos atributos “c” e “u”.

Exemplos:

- chattr +AacdiSsu teste.txt - Adiciona todos os atributos
- chattr =ASs teste.txt - Define os atributos para “ASs”
- chattr +i -A teste.txt - Retira o atributo “A” e adiciona “i”

- chattr = teste.txt - Retira todos os atributos

1.29 lsattr

Lista atributos de um arquivo/diretório. Os atributos podem ser modificados através do comando chattr.

lsattr [opções] [arquivos/diretórios]

Onde:

arquivos/diretórios

Arquivos/diretórios dos quais se deseja listar os atributos. Podem ser usados curingas.

opções

-a

Lista todos os arquivos, incluindo ocultos (iniciando com um ".").

-d

Lista os atributos de diretórios, ao invés de listar os arquivos que ele contém.

-R

Faz a listagem em diretórios e subdiretórios.

-v

Mostra versões dos arquivos.

Caso seja especificado sem parâmetros, o lsattr listará os atributos de todos os arquivos e diretórios do diretório atual. O lsattr mostrará mensagens de erro caso seja usado em um diretório de pontos de montagem ou arquivos que não sejam ext2.

Exemplo: lsattr -d, lsattr -R, lsattr -R *.txt

1.30 cut

Mostra seções de cada linha do arquivo, dependendo das opções passadas ao programa.

cut [opções] [arquivo]

Onde:

arquivo

Arquivo que será verificado pelo comando cut.

opções

-b, --bytes [bytes]

Mostra somente a lista de [bytes] do arquivo.

-c, --characters [numero]

Mostra somente o [número] de caracteres no arquivo. É semelhante a opção -b mas tabs e espaços são tratados como qualquer caracter.

-f, --field [campos]

Mostra somente a lista de [campos].

-d, --delimite [delimitador]

Para uso com a opção -f, os campos são separados pelo primeiro caracter em [delimitador] ao invés de tabulações.

-s

Para uso com a opção -f, somente mostra linhas que contém o caracter separador de campos.

- Devem ser especificadas opções para o funcionamento deste comando. Os bytes, campos e delimitadores podem ser especificados através de intervalos de caracteres (usando a-z), através de vírgulas (a,b,d) ou da combinação entre eles. cut -b 1,3 /etc/passwd - Pega a primeira e terceira letra (byte) de cada linha do arquivo /etc/passwd
- cut -b 1,3-10 /etc/passwd - Pega a primeira letra (byte) e terceira a décima letra de cada linha do arquivo /etc/passwd.
- cut -c 1,3-10 /etc/passwd - Pega o primeiro caractere e terceiro ao décimo caractere de cada linha do arquivo /etc/passwd.

1.31 cmp

Compara dois arquivos de qualquer tipo (binário ou de texto). Caso exista diferença entre eles, é mostrado o número da linha e byte onde ocorreu a primeira diferença na saída padrão (tela) e o programa retorna o código de saída 1.

cmp [arquivo1] [arquivo2] [opções]

Opções:

arquivo1/arquivo2

Arquivos que serão comparados.

opções

-l

Mostra o número do byte (hexadecimal) e valores diferentes de bytes (octal) para cada diferença.

-s

Não mostra nenhuma diferença, só retorna o código de saída do programa.

Use o comando zcmp para comparar diretamente arquivos binários ou de texto compactados com gzip.

Exemplo: cmp teste.txt teste1.txt.

1.32 dirname

Obtém o nome do diretório através do caminho passado ao programa.

dirname [diretório/arquivo]

Exemplos: dirname /usr/bin/dirname, dirname /tmp/*.

1.33 diff

Compara dois arquivos e mostra as diferenças entre eles. O comando diff é usado somente para a comparação de arquivos em formato texto. As diferenças encontradas podem ser redirecionadas para um arquivo que poderá ser usado pelo comando patch, a fim de aplicar as alterações num arquivo que não contenha as diferenças. Isso é útil para grandes textos, porque é possível copiar somente as modificações (geradas através do diff, que são muito pequenas) e aplicar ao arquivo para atualizá-lo (através do patch), ao invés de copiar a nova versão. Esse é um sistema de atualização muito usado na atualização dos códigos-fonte do kernel do GNU/Linux.

diff [diretório1/arquivo1] [diretório2/arquivo2] [opções]

Opções:

diretório1/arquivo1 diretório2/arquivo2

Arquivos/diretórios que serão comparados. Normalmente é usado como primeiro arquivo/diretório o mais antigo e o mais novo como segundo.

opções

-lines [num]

Gera a diferença com [num] linhas de contexto. Por padrão, o diff gera um arquivo com duas linhas, que é o mínimo necessário para o correto funcionamento do patch.

-a

Compara os dois arquivos como arquivos texto.

-b

Ignora espaços em branco como diferenças.

-B

Ignora linhas em branco inseridas ou apagadas nos arquivos.

-i

Ignora diferenças entre maiúsculas e minúsculas nos arquivos.

-H

Usa análise heurística para verificar os arquivos.

-N

Em uma comparação de diretórios, se o arquivo apenas existe em um diretório, trata-o como presente mas vazio no outro diretório.

-P

Em uma comparação de diretórios, se o arquivo apenas existe no segundo diretório, trata-o como presente mas vazio no primeiro diretório.

-q

Mostra somente se os dois arquivos possuem diferenças. Não mostra as diferenças entre eles.

-r

Compara diretórios e sub-diretórios existentes.

-S [nome]

Inicia a comparação de diretórios pelo arquivo [nome]. É útil quando cancelamos uma comparação.

-t

Aumenta a tabulação das diferenças encontradas.

-u

Usa o formato de comparação unificado.

Use o comando zdiff para comparar diretamente arquivos compactados pelo utilitário gzip

Use o comando sdiff para visualizar as linhas diferentes entre os dois arquivos em formato texto simples.

- Exemplodiff texto.txt texto1.txt - Compara o arquivo texto.txt com texto1.txt e exibe suas diferenças na tela.
- diff -Bu texto.txt texto1.txt - Compara o arquivo texto.txt com texto1.txt ignorando linhas em branco diferentes entre os dois arquivos e usando o formato unificado.
- diff texto.txt texto1.txt >texto.diff - Compara o arquivo texto.txt com texto1.txt e gera um arquivo chamado texto. diff contendo a diferença entre eles. Este arquivo poderá ser usado pelo patch para aplicar as diferenças existente entre os dois no arquivo texto.txt.
- diff -r /usr/src/linux-2.2.13 /usr/src/linux-2.2.14 >patch-2.2.14.diff - Compara o diretório e sub-diretórios linux-2.2.13 e linux-2.2.14 e grava as diferenças entre eles no arquivo patch-2.2.14.diff.

1.34 pr

Pagina arquivos texto ou a entrada padrão para impressão. Este comando faz a paginação de um arquivo texto e opcionalmente ajusta o número de colunas e mostra o resultado na saída padrão.

pr [*opções*] [*arquivo*]

Onde:

arquivo

Arquivo que será paginado para impressão.

opções

+[NUM]

Inicia a numeração de páginas na página [PAGINA]

-[NUM]

Mostra a saída com [NUM] colunas.

-c

Imprime o caracter CTRL como “^” na saída padrão.

-F, -f

Usa avanço de página ao invés de linhas em branco para separar páginas.

-e[caracter][tamanho]

Usa o caracter [caracter] como tabulação (o padrão é TAB) e o espaço da tabulação [tamanho].

-h [nome]

Mostra [nome] ao invés do nome do arquivo no cabeçalho.

-l [num]

Define o número máximo de linhas por página para [num].

-m

Imprime vários arquivos em paralelo, um por coluna.

-r

Oculta mensagens de erro de abertura de arquivos.

-w [num]

Ajusta a largura da página para [num] colunas (o padrão é 72).

Exemplo: pr -l 50 -h “Teste do comando pr” teste.txt.

1.35 patch

Atualiza arquivos texto através das diferenças geradas pelo comando diff.

patch [opções] [arquivo.diff] ou

patch [opções] < [arquivo.diff]

Onde:

arquivo.diff

Arquivo contendo as diferenças geradas pelo comando diff.

opções

-p [num]

Nível do diretório em que o patch será aplicado. Se for igual a 0, o patch assume que os arquivos que serão atualizados estão no diretório atual, se 1, assume que os arquivos que serão atualizado estão no diretório acima (..), se 2, 2 diretórios acima ...

-b

Cria cópias de segurança dos arquivos originais ao aplicar o patch.

-binary

Lê e grava arquivo usando modo binário.

-d [dir]

Muda para o diretório [dir] antes de aplicar o patch.

-E

Remove arquivos vazios após a aplicação do patch.

-n

Interpreta o arquivo de patch como um .diff normal.

-N

Não desfaz patches já aplicados.

-s

Não mostra mensagens de erro.

-u

Interpreta o patch em formato unificado.

As diferenças são aplicadas em arquivos originais gerados pelo comando *diff.* É importante entender os comandos patch e diff, pois são comandos muito utilizados para desenvolvimento feito por equipes de pessoas.

- Exemplo:patch -p0<texto.diff - Aplica as diferenças contidas no arquivo texto.diff nos arquivos originais.
- patch -p0 texto.txt texto.diff - Aplica as diferenças contidas no arquivo texto.diff nos arquivos originais. Faz a mesma coisa que o comando anterior.

1.36 whereis

Localiza o arquivo que contém uma página de manual. A pesquisa é feita usando os caminhos de páginas de manuais configuradas no sistema (normalmente o arquivo /etc/manpath.config).

whereis [comando]

Exemplo: whereis ls, whereis cd.

1.37 which

Mostra a localização de um arquivo executável no sistema. A pesquisa de arquivos executáveis é feita através do path do sistema.

which [comando]

Exemplos: which ls, which shutdown, which which.

1.38 zforce

Renomeia extensão de arquivos para .gz. Esse comando é útil quando fazemos downloads de arquivos compactados pelo gzip, mas que não estejam identificados pela extensão .gz.

zforce [arquivos]

Quando é usado o zforce, é importante verificar se o arquivo é um arquivo compactado pelo gzip. Caso seja, é verificado se já existe a extensão .gz. Se ela não estiver lá, acrescente-a.

1.39 gzexe

Cria arquivos compactados gzip auto-extraíveis. Este comando é usado para compactar arquivos executáveis que se auto-descompactam assim que solicitado. É útil para sistemas ou unidades de disco que possuem pouco espaço disponível. Deve ser usado somente para arquivos executáveis.

gzexe [arquivo]

Onde: *arquivo* é o arquivo executável que será compactado.

Quando gzexe é executado, uma cópia do arquivo original é gravada com o formato nome_do_arquivo~.

Exemplo: gzexe /tmp/teste.

1.40 znew

Recompacta arquivos do formato compress (.Z) para o formato gzip (.gz). Após a recompactação, os arquivos de origem .Z são apagados.

znew [opções] [arquivo]

Onde:

arquivo.Z

Arquivo compactado pelo compress, que será recompactado para o gzip.

opções

-f

Substitui o arquivo .gz caso já exista.

-t

Teste os novos arquivos criados antes de apagar os arquivos .Z.

-v

Mostra o nome e porcentagem de compactação para cada arquivo processado.

-9

Usa a máxima compactação.

-P

Usa pipes durante a conversão para reduzir o espaço ocupado no disco. Caso esta opção seja usada, a data e a hora do arquivo não são mantidas.

-K

Mantém o arquivo .Z caso seja menor que o arquivo .gz.

1.41 tr

O comando tr pode ser entendido como *translate* (traduzir), pois é exatamente o que ele faz. Recebe como entrada um texto e retorna a entrada modificada de acordo com os parâmetros passados ao comando. Por exemplo:

```
$ echo Isso e um teste
Isso e um teste
(observe a crase (‘) e o piper (|) que acompanha o comando)

$ echo Isso e um teste | tr ‘a-z’ ‘A-Z’
ISSO E UM TESTE

$ echo Isso e um teste |tr -d ‘ ‘
Issoeumteste

$ echo Isso e um teste |tr ‘ ‘ ‘\t’
Isso e um teste
```

Capítulo 02 - Permissões de acesso a arquivos e diretórios

A permissão de acesso protege o sistema de arquivos GNU/Linux do acesso indevido de pessoas ou programas não autorizados.

A permissão de acesso do GNU/Linux também impede que um programa mal intencionado, por exemplo, apague um arquivo que não deve, envie arquivos para outra pessoa ou forneça acesso da rede para que outros usuários invadam o sistema. O sistema GNU/Linux é muito seguro e, como qualquer outro sistema seguro e confiável, impede que usuários iniciantes (ou mal intencionados) instalem programas enviados por terceiros - sem saber para que eles realmente sirvam - e causem danos irreversíveis em seus arquivos, seu micro ou sua empresa.

Por esta seção pode se tornar um pouco difícil de se entender, recomendamos praticá-la ao mesmo tempo em que ela for lida, para uma ótima compreensão. Mas não se preocupe, também há exemplos para ajudá-lo a entender o sistema de permissões de acesso do ambiente GNU/Linux.

2.1 Donos, grupos e outros usuários

O princípio da segurança no sistema de arquivos GNU/Linux é definir o acesso aos arquivos por donos, grupos e outros usuários:

dono

É a pessoa que criou o arquivo ou o diretório. O nome do

dono do arquivo/diretório é o mesmo do usuário usado para entrar no sistema GNU/Linux. Somente o dono pode modificar as permissões de acesso do arquivo.

As permissões de acesso do dono de um arquivo somente se aplicam a ele próprio. A identificação do dono também é chamada de user id (UID).

A identificação de usuário e o nome do grupo a que pertence são armazenados respectivamente nos arquivos /etc/passwd e /etc/group. Estes são arquivos textos comuns e podem ser editados em qualquer editor de texto. Mas tenha cuidado para não modificar o campo que contém a senha do usuário encriptada (que pode estar armazenada neste arquivo, se não estiver usando senhas ocultas).

grupo

Para permitir que vários usuários diferentes tenham acesso a um mesmo arquivo (já que somente o dono pode ter acesso ao arquivo), este recurso foi criado. Cada usuário pode fazer parte de um ou mais grupos e então acessar arquivos que pertençam ao mesmo grupo que o seu (mesmo que esses arquivos tenham outro *dono*).

Por padrão, quando um novo usuário é criado, o grupo ao qual ele pertencerá será o mesmo de seu grupo primário (exceto pelas condições que explicarei adiante). A identificação do grupo é chamada de group id (GID).

Um usuário pode pertencer a um ou mais grupos. Mais detalhes serão vistos adiante.

outros

É a categoria de usuários que não são donos ou não pertencem ao grupo do arquivo.

Cada um dos tipos acima possui três tipos básicos de permissões de acesso, que serão vistas na próxima seção.

2.2 Tipos de Permissões de acesso

Quanto aos tipos de permissões que se aplicam ao *dono*, *grupo* e *outros usuários*, temos 3 permissões básicas:

- r - Permissão de leitura para arquivos. Caso seja um diretório, permite listar seu conteúdo (através do comando ls, por exemplo).

- w - Permissão de gravação para arquivos. Se for um diretório, permite a gravação de arquivos ou outros diretórios dentro dele.

Para que um arquivo/diretório possa ser apagado, é necessário o acesso a gravação.

- x - Permite executar um arquivo (caso seja um programa executável). Se for um diretório, permite que seja acessado através do comando cd.

As permissões de acesso a um arquivo/diretório podem ser visualizadas com o uso do comando ls -la, como já explicado neste material. As 3 letras (rwx) são agrupadas da seguinte forma:

- ***rwxrwxrwx*** gleydson users teste

Virou uma bagunça, não? Vou explicar cada parte para entender o que querem dizer as 10 letras acima (da esquerda para a direita):

- Primeira letra: diz qual é o tipo do arquivo. Se for um d é um diretório; um l um link a um arquivo no sistema; um - quer dizer que é um arquivo comum, etc.
- Segunda à quarta letra: (rwx) representam a permissão de acesso ao *dono* do arquivo. Neste caso *gleydson* tem a permissão de ler (r - read), gravar (w - write) e executar (x - execute) o arquivo teste.
- Quinta à sétima letra: (rwx) dizem respeito à permissão de acesso ao *grupo* do arquivo. Neste caso, todos os usuários que pertencem ao grupo *users* têm a permissão de ler (r), gravar (w), e também executar (x) o arquivo teste.
- Três últimas letras: (rwx) indicam a permissão de acesso para os *outros usuários*. Neste caso, todos os usuários que não são donos do arquivo teste ou que não pertençam ao grupo users têm a permissão para ler, gravar e executar o programa.

O comando *chmod* é utilizado para alterar as permissões de acesso de arquivos ou diretórios, como veremos mais à frente.

2.3 Etapas para acesso a um arquivo/diretório

O acesso a um arquivo/diretório é feito verificando, antes de tudo, se o usuário que o acessará é o seu *dono*. Caso seja, as permissões de dono do arquivo são aplicadas. Caso não seja o *dono* do arquivo/diretório, é verificado se ele pertence ao grupo correspondente. Caso pertença, as permissões do *grupo* são aplicadas. Caso não pertença ao *grupo*, são verificadas as permissões de acesso para os outros usuários que não são *donos* e não pertençam ao *grupo* correspondente ao arquivo/diretório.

Após verificar onde o usuário se encaixa nas permissões de acesso do arquivo (se ele é o *dono*, pertence ao *grupo*, ou *outros usuários*), é verificado se ele terá permissão para o que deseja fazer (ler, gravar ou executar o arquivo). Se não tiver, o acesso é negado, mostrando uma mensagem do tipo: “Permission denied” (permissão negada).

Isso quer dizer que mesmo você sendo o dono do arquivo e definir o acesso do *dono* (através do comando chmod) como somente leitura (r) mas o acesso dos *outros usuários* como leitura e gravação, você somente poderá ler esse arquivo, mas os outros usuários poderão lê-lo ou gravar nele.

As permissões de acesso (leitura, gravação, execução) para donos, grupos e outros usuários são independentes, permitindo assim um nível de acesso diferenciado.

Lembre-se: Somente o dono pode modificar as permissões de acesso de um arquivo ou diretório!

2.4 Exemplos práticos de permissões de acesso

Seguem dois exemplos práticos de permissão de acesso: a um arquivo e a um diretório. Eles serão explicados passo-a-passo para uma perfeita compreensão do assunto. Vamos à prática!

2.4.1 Exemplo de acesso a um arquivo

Segue abaixo um exemplo de acesso a um arquivo no GNU/Linux (obtido com o comando ls -la explicado passo a passo:

-rwxr-xr-- 1 gleydson user 8192 nov 4 16:00 teste

-rwxr-xr--

Essas são as permissões de acesso ao arquivo teste. Um conjunto de dez letras que especificam o tipo do arquivo, permissão do dono do arquivo, grupo do arquivo e outros usuários. Veja a explicação detalhada sobre cada uma abaixo:

-rwxr-xr--

A primeira letra (do conjunto das 10) determina o tipo do arquivo. Se a letra for um d, é um diretório, e você poderá acessá-lo usando o comando cd. Caso for um l é um link simbólico para algum arquivo ou diretório no sistema. Um - significa que é um arquivo normal.

-rwxr-xr--

Essas três letras (da segunda à quarta do conjunto de 10) são as permissões de acesso do dono do arquivo teste. O dono (neste caso gleydson) tem a permissão para ler (r), gravar (w) e executar (x) o arquivo teste.

-rwxr-xr--

Essas três letras (da quinta à sétima do conjunto) são as permissões de acesso dos usuários que pertencem ao grupo user do arquivo teste. Os usuários que pertencem ao grupo user têm a permissão somente para ler (r) e executar (x) o arquivo teste, não podendo modificá-lo ou apagá-lo.

-rwxr-xr--

Essas 3 letras (as últimas) são as permissões de acesso para usuários que não são o dono do arquivo teste e que não pertencem ao grupo user. Nesse caso, essas pessoas somente terão a permissão para ver (r – read) o conteúdo do arquivo teste.

gleydson

Nome do dono do arquivo teste.

user

Nome do grupo ao qual o arquivo teste pertence.

teste

Nome do arquivo.

2.4.2 Exemplo de acesso a um diretório

Abaixo temos um exemplo com explicações das permissões de acesso a um diretório no GNU/Linux:

---

drwxr-x--- 2 gleydson user 1024 nov 4 17:55 exemplo

drwxr-x---

Permissões de acesso ao diretório exemplo. É um conjunto de dez letras que especificam o tipo de arquivo, permissão do dono do diretório, grupo ao qual o diretório pertence e permissão de acesso a outros usuários. Veja as explicações abaixo:

drwxr-x---

A primeira letra (do conjunto das dez) determina o tipo do arquivo. Nesse caso é um diretório, pois tem a letra d.

drwxr-x---

Essas 3 letras (da segunda à quarta) são as permissões de acesso do dono do diretório exemplo. O dono do diretório (neste caso gleydson) tem a permissão para listar arquivos do diretório (r), gravar arquivos no diretório (w) e entrar no diretório (x).

drwxr-x---

Essas 3 letras (quinta, sexta e sétima) são as permissões de acesso dos usuários que pertencem ao grupo user. Os usuários que pertencem ao grupo user têm a permissão somente para listar arquivos do diretório (r) e entrar no diretório (x) exemplo.

drwxr-x---

Essas 3 letras (últimas do conjunto de 10) são as permissões de acesso para usuários que não são dono do diretório exemplo e que não pertencem ao grupo user. Com as permissões acima, nenhum usuário que se encaixe nas condições de dono e grupo do diretório têm permissão para acessá-lo.

gleydson

Nome do dono do diretório exemplo.

user

Nome do grupo ao qual o diretório exemplo pertence.

exemplo

Nome do diretório.

- **OBSERVAÇÕES:** usuário root não tem nenhuma restrição de acesso ao sistema.

- Se você possui permissões de gravação no diretório e tentar apagar um arquivo a que não tenha permissão de gravação, o sistema perguntará por confirmar a exclusão do arquivo, apesar do modo leitura. Caso você tenha permissões de gravação no arquivo, o arquivo será apagado por padrão sem mostrar nenhuma mensagem de erro (a não ser que seja especificada a opção -i com o comando rm).

- Por outro lado, mesmo que você tenha permissões de gravação em um arquivo mas não tenha permissões de gravação em um diretório, a exclusão do arquivo será negada.

Isso mostra que a permissão de acesso do diretório é levada mais em consideração do que as permissões dos arquivos e sub-diretórios que ele contém. Esse ponto é muitas vezes ignorado por muitas pessoas, mas tome cuidado, pois ele expõe seus sistemas a riscos de segurança. Imagine o problema de um usuário que não tenha permissão de gravação num arquivo, mas que tenha no diretório, pode causar em um sistema mal administrado.

2.5 Permissões de Acesso Especiais

Em adição às três permissões básicas (rwx), existem permissões de acesso especiais (stX) que afetam arquivos executáveis e diretórios:

- s - Quando é usado na permissão de acesso do d*ono*, ajusta a identificação efetiva do usuário do processo durante a execução de um programa. É também chamado de *bit setuid* e não tem efeito em diretórios.

  Quando s é usado na permissão de acesso do g*rupo*, ajusta a identificação efetiva do grupo do processo durante a execução de um programa, chamado de *bit setgid*. É identificado pela letra s no lugar da permissão de execução do grupo do arquivo/diretório. Em diretórios, força que os arquivos criados dentro dele pertençam ao mesmo grupo do diretório, ao invés do grupo primário que o usuário pertence.

  Setgid e *setuid* podem aparecer ao mesmo tempo, no mesmo arquivo/diretório. A permissão de acesso especial s somente pode aparecer no campo *dono* e *grupo*.

- S - Idêntico a “s”. Significa que não existe a permissão “x” (execução ou entrar no diretório) naquele lugar. Um exemplo é o chmod 2760 em um diretório.

- t - Salva a imagem do texto do programa no dispositivo swap. Assim ele será carregado mais rapidamente quando executado. Também é chamado de *stick bit*.

  Em diretórios, impede que outros usuários removam arquivos dos quais não são donos. Isso é chamado de colocar o diretório em modo append-only. Um exemplo de diretório que se encaixa perfeitamente nesta condição é o /tmp. Todos os usuários devem ter acesso para que seus programas possam criar os arquivos temporários lá, mas nenhum pode apagar arquivos dos outros. A permissão especial t pode

ser especificada somente no campo outros usuários das permissões de acesso.

- T - Idêntico a “t”. Significa que não existe a permissão “x” naquela posição (por exemplo, em um chmod 1776 em um diretório).
- X - Se você usar X ao invés de x, a permissão de execução somente é afetada se o arquivo já tiver permissões de execução. Em diretórios ela tem o mesmo efeito que a permissão de execução x.

Exemplo da permissão de acesso especial X:

1. *Crie um arquivo teste (digitando touch teste) e defina sua permissão para rw-rw-r-- (chmod ug=rw,o=r teste ou chmod 664 teste).*
2. *Agora use o comando chmod a+X teste*
3. *digite ls -l*
4. *Veja que as permissões do arquivo não foram afetadas.*
5. *agora digite chmod o+x teste*
6. *digite ls -l, você colocou a permissão de execução para os outros usuários.*
7. *Agora use novamente o comando chmod a+X teste*
8. *digite ls -l*
9. *Veja que agora a permissão de execução foi concedida a todos os usuários, pois foi verificado que o arquivo era executável (tinha permissão de execução para outros usuários).*
10. *Agora use o comando chmod a-X teste*
11. *Ele também funcionará e removerá as permissões de execução de todos os usuários, porque o arquivo teste tem permissão de execução (confira digitando ls -l).*
12. *Agora tente novamente o chmod a+X teste*
13. *Você deve ter reparado que a permissão de acesso especial X é semelhante a x, mas somente faz efeito quanto o arquivo já tem permissão de execução para o dono, grupo ou outros usuários.*

Em diretórios, a permissão de acesso especial X funciona da mesma forma que x, até mesmo se o diretório não tiver nenhuma permissão de acesso (x).

2.6 A conta root

A conta root é também chamada de *super usuário*. É um login que não possui restrições de segurança. A conta root somente deve ser usada para fazer a administração do sistema, e usada durante o

menor período de tempo possível.

Qualquer senha que criar deverá conter de 6 a 8 caracteres (em sistemas usando crypto) ou até frases inteiras (caso esteja usando MD5, que garante maior segurança), e também poderá conter letras maiúsculas e minúsculas, além de caracteres de pontuação. Tenha um cuidado especial quando escolher sua senha root, porque ela é a conta mais poderosa. Evite palavras de dicionário ou o uso de quaisquer outros dados pessoais que possam ser adivinhados.

Se qualquer um lhe pedir senha root, seja extremamente cuidadoso. Você normalmente nunca deve distribuir sua conta root, a não ser que esteja administrando um computador com mais de um administrador do sistema.

Utilize uma conta de usuário normal ao invés da conta root para operar seu sistema. Por que não usar a conta root? Bem, uma razão para evitar usar privilégios root é a facilidade de se cometer danos irreparáveis ao sistema. Outra razão é que você pode ser enganado e rodar um programa do tipo *Cavalo de Tróia*, **que obtém poderes do *super usuário* para comprometer a segurança do seu sistema sem que você saiba.**

2.7 chmod

Muda a permissão de acesso de um arquivo ou diretório. Com este comando você pode escolher se um usuário ou grupo terá permissão para ler, gravar ou executar arquivos. Sempre que um arquivo é criado, seu dono é o usuário que o criou e seu grupo é o grupo de tal usuário (exceto para diretórios configurados com a permissão de grupo s).

chmod [opções] [permissões] [diretório/arquivo]

Onde:

diretório/arquivo

Diretório ou arquivo que terá sua permissão alterada.

opções

-v, --verbose

Mostra todos os arquivos que estão sendo processados.

-f, --silent

Não mostra a maior parte das mensagens de erro.

-c, --change

Semelhante à opção -v, mas só mostra os arquivos que tiveram as permissões alteradas.

-R, --recursive

Muda permissões de acesso do *diretório/arquivo* no diretório atual e sub-diretórios.

- **ugoa**+-=rwxXs*ugoa* - Controla que nível de acesso será mudado. Especificam, em ordem, usuário (u), grupo (g), outros (o), todos (a).

- **+-=**: + coloca a permissão, - retira a permissão do arquivo e = define a permissão exatamente como especificado.

- rwx - *r* permissão de leitura do arquivo. *w* permissão de gravação. *x* permissão de execução (ou acesso a diretórios).

chmod não muda permissões de links simbólicos. Elas devem ser mudadas no arquivo alvo do link. Também podem ser usados códigos numéricos octais para a mudança das permissões de acesso a arquivos/diretórios. Veja mais detalhes adiante.

DICA: É possível copiar permissões de acesso do arquivo/diretório, por exemplo, se o arquivo teste.txt tiver a permissão de acesso r-xr----- e você digitar chmod o=u, as permissões de acesso dos outros usuários (o) serão idênticas ao do dono (u). Então a nova permissão de acesso do arquivo teste.txt será r-xr-r-x.

Exemplos de permissões de acesso:

- chmod g+r * - Permite que todos os usuários que pertençam ao grupo dos arquivos (g) tenham (+) permissões de leitura (r) em todos os arquivos do diretório atual.
- chmod o-r teste.txt - Retira (-) a permissão de leitura (r) do arquivo teste.txt para os outros usuários (usuários que não são donos e não pertencem ao grupo do arquivo teste.txt).
- chmod uo+x teste.txt - Inclui (+) a permissão de execução do arquivo teste.txt para o dono e outros usuários do arquivo.
- chmod a+x teste.txt - Inclui (+) a permissão de execução do arquivo teste.txt para o dono, grupo e outros usuários.
- chmod a=rw teste.txt - Define a permissão de todos os usuários exatamente (=) para leitura e gravação do arquivo teste.txt.

2.8 chgrp

Muda o grupo de um arquivo/diretório.

chgrp [opções] [grupo] [arquivo/diretório]

Onde:

grupo

Novo grupo do *arquivo/diretório*.

arquivo/diretório

Arquivo/diretório que terá o grupo alterado.

opções

-c, --changes

Somente mostra os arquivos/grupos que forem alterados.

-f, --silent

Não mostra mensagens de erro para arquivos/diretórios que não puderam ser alterados.

-v, --verbose

Mostra todas as mensagens e arquivos sendo modificados.

-R, --recursive

Altera os grupos de arquivos e sub-diretórios do diretório

atual.

2.9 chown

Muda dono de um arquivo/diretório. Opcionalmente pode também ser usado para mudar o grupo.

chown [opções] [dono.grupo] [diretório/arquivo]

Onde:

dono.grupo

Nome do *dono* e do *grupo* que será atribuído ao *diretório/arquivo*. O grupo é opcional.

diretório/arquivo

Diretório/arquivo para o qual o *dono.grupo* **será modificado.**

opções

-v, --verbose

Mostra os arquivos enquanto são alterados.

-f, --supress

Não mostra mensagens de erro durante a execução do programa.

-c, --changes

Mostra somente arquivos que foram alterados.

-R, --recursive

Altera dono e grupo de arquivos no diretório atual e sub-diretórios.

O *dono.grupo* pode ser especificado usando o nome de grupo ou o código numérico correspondente ao grupo (GID).

- Você deve ter permissões de gravação no diretório/arquivo para alterar seu dono/grupo.chown joao teste.txt - Muda o dono do arquivo teste.txt para joao.
- chown joao.users teste.txt - Muda o dono do arquivo teste.txt para joao e seu grupo para users.
- chown -R joao.users * - Muda o dono/grupo dos arquivos do diretório atual e sub-diretórios para joao/users (desde que você tenha permissões de gravação no diretórios e sub-diretórios).

2.10 Modo de permissão octal

Ao invés de utilizar os modos de permissão +r, -r, etc, pode ser usado o modo octal para se alterar a permissão de acesso a um arquivo. O modo octal é um conjunto de oito números, onde cada um deles define um tipo de acesso diferente.

- É mais flexível gerenciar permissões de acesso usando o modo octal ao invés do comum, pois você especifica diretamente a permissão do dono, grupo, outros ao invés de gerenciar as permissões de cada um separadamente. Veja abaixo a lista de permissões de acesso octal: 0 - Nenhuma permissão de acesso. Equivalente a -rwx.
- 1 - Permissão de execução (x).

- 2 - Permissão de gravação (w).
- 3 - Permissão de gravação e execução (wx).
- 4 - Permissão de leitura (r).
- 5 - Permissão de leitura e execução (rx).
- 6 - Permissão de leitura e gravação (rw).
- 7 - Permissão de leitura, gravação e execução.

(Equivalente a +rwx).

O uso de um desses números define a permissão de acesso do *dono*, *grupo* ou *outros usuários*. Um modo fácil de entender como as permissões de acesso octais funcionam é através da seguinte tabela:

1 = Executar

2 = Gravar

4 = Ler

* Para Dono e Grupo, multiplique as permissões acima por x100 e x10.

e para as permissões de acesso especiais:

1000 = Salva imagem do texto no dispositivo de troca

2000 = Ajusta o bit setgid na execução

4000 = Ajusta o bit setuid na execução

Basta, agora, fazer o seguinte:

- Somente permissão de execução, use 1.
- Somente a permissão de leitura, use 4.
- Somente permissão de gravação, use 2.
- Permissão de leitura/gravação, use 6 (equivale a 2+4 / Gravar+Ler).
- Permissão de leitura/execução, use 5 (equivale a 1+4 / Executar+Ler).
- Permissão de execução/gravação, use 3 (equivale a 1+2 / Executar+Gravar).
- Permissão de leitura/gravação/execução, use 7 (equivale a 1+2+4 / Executar+Gravar+Ler).
- Salvar texto no dispositivo de troca, use 1000.
- Ajustar bit setgid, use 2000.
- Ajustar bit setuid, use 4000.
- Salvar texto e ajustar bit setuid, use 5000 (equivale a 1000+4000 / Salvar texto + bit setuid).
- Ajustar bit setuid e setgid, use 6000 (equivale a 4000+2000 / setuid + setgid).

Vamos à prática com alguns exemplos:

chmod 764 teste

Os números são interpretados da **direita para a esquerda**

como permissão de acesso aos *outros usuários* (4), *grupo* (6), e *dono* (7). O exemplo acima faz os *outros usuários* (4) terem acesso somente leitura (r) ao arquivo teste, o *grupo* (6) ter a permissão de leitura e gravação (rw), e o *dono* (7) ter permissão de leitura, gravação e execução (rwx) ao arquivo teste.

chmod 40 teste

O exemplo acima define a permissão de acesso dos *outros usuários* (0) como nenhuma, e define a permissão de acesso do *grupo* (4) como somente leitura (r). Note: usei somente dois números e então a permissão de acesso do *dono* do arquivo **não** é modificada (leia as permissões de acesso da direita para a esquerda!). Para mais detalhes, veja a lista de permissões de acesso em modo octal, no início desta seção.

chmod 751 teste

O exemplo acima define a permissão de acesso dos *outros usuários* (1) para somente execução (x), o acesso do *grupo* (5) como leitura e execução (rx) e o acesso do *dono* (7) como leitura, gravação e execução (rwx).

chmod 4751 teste

O exemplo acima define a permissão de acesso dos *outros usuários* (1) para somente execução (x), acesso do *grupo* (5) como leitura e execução (rx), o acesso do *dono* (7) como leitura, gravação e execução (rwx) e ajusta o bit setgid (4) para o arquivo teste.

2.11 umask

A umask (*user mask*) são três números que definem as permissões iniciais do *dono*, *grupo* e *outros usuários* que o arquivo/diretório receberá quando for criado ou copiado. Digite *umask* sem parâmetros para retornar ao valor de sua umask atual.

A umask tem efeitos diferentes caso o arquivo que estiver sendo criado for binário (um programa executável) ou de texto. Veja a tabela a seguir para ver qual é a mais adequada a sua situação:

UMASK	ARQUIVO		DIRETÓRIO
	Binário	Texto	
0	r-x	rw-	rwx
1	r--	rw-	rw-
2	r-x	r--	r-x
3	r--	r--	r--
4	--x	-w-	-wx
5	---	-w-	-w-
6	--x	---	--x
7	---	---	---

+-------+-----------+---------+-----------+

Um arquivo texto criado com o comando umask 012;touch texto.txt receberá as permissões -rw-rw-r--, pois 0 (dono) terá permissões rw-, 1 (grupo), terá permissões rw- e 2 (outros usuários) terão permissões r--. Um arquivo binário copiado com o comando umask 012;cp /bin/ls /tmp/ls receberá as permissões -r-xr--r-x (confira com a tabela acima).

Por esse motivo é preciso um pouco de atenção antes de escolher a umask, pois um valor mal escolhido pode causar problemas de acesso a arquivos, diretórios ou programas não sendo executados. Na maioria das distribuições atuais, o valor padrão da umask é 022. A umask padrão no sistema Debian é a 022.

A umask é de grande utilidade para programas que criam arquivos/diretórios temporários. Sendo assim, pode-se bloquear o acesso de outros usuários desde a criação do arquivo, evitando recorrer ao chmod.