LAB – PIPES, REDIRECIONAMENTO E REGEX

8.1 Introdução

Este é o Lab 8: Pipes, Redirecionamento e REGEX. Ao realizar este laboratório, os alunos aprenderão como redirecionar fluxos de texto, usar expressões regulares e comandos para filtrar arquivos de texto.

Neste laboratório, você executará as seguintes tarefas:

- Aprender a redirecionar e utilizar pipes nos canais padrões de entrada, saída e erro.
- Usar expressões regulares para filtrar a saída de comandos ou conteúdo de arquivo.
- Exibir arquivos grandes ou saída de comando com programas para paginação e visualização de partes selecionadas.

8.2 Pipes para linha de comando e redirecionamento

Normalmente, quando você executa um comando, a saída é exibida na janela do terminal. Essa saída (também chamada de canal) é chamada de saída padrão, simbolizada pelo termo `stdout`. O número do descritor de arquivo para este canal é 1.

Erro padrão (`stderr`) ocorre quando ocorre um erro durante a execução de um comando; ele tem um descritor de arquivo de 2. Mensagens de erro também são enviadas para a janela do terminal por padrão.

Neste laboratório, você usará caracteres que redirecionam a saída da saída padrão (`stdout`) e o erro padrão (`stderr`) para um arquivo ou para outro comando, em vez da tela do terminal.

A entrada padrão, `stdin`, geralmente é fornecida por você para um comando, digitando no teclado; ele tem um descritor de arquivo de 0. No entanto, redirecionando a entrada padrão, os arquivos também podem ser usados como `stdin`.

8.2.1 Passo 1

Use o símbolo de redirecionamento `>` para redirecionar a saída da saída normal de `stdout` (terminal) para um arquivo. Digite o seguinte:

```
echo "Hello World"
echo "Hello World" > mymessage
cat mymessage
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ echo "Hello World"
Hello World
sysadmin@localhost:~$ echo "Hello World" > mymessage
sysadmin@localhost:~$ cat mymessage
Hello World
sysadmin@localhost:~$
```

O primeiro comando ecoa a mensagem (`stdout`) para o terminal.

O segundo comando redireciona a saída; em vez de enviá-lo para o terminal, a saída é enviada para um arquivo chamado `mymessage`.

O último comando exibe o conteúdo do arquivo `mymessage`.

8.2.2 Passo 2

Quando você usa o símbolo > para redirecionar o `stdout`, o conteúdo do arquivo é primeiro destruído. Digite os seguintes comandos para ver uma demonstração:

```
cat mymessage
echo Greetings > mymessage
cat mymessage
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ cat mymessage
Hello World
sysadmin@localhost:~$ echo Greetings > mymessage
sysadmin@localhost:~$ cat mymessage
Greetings
sysadmin@localhost:~$
```

Observe que usar um símbolo de redirecionamento sobrescreve um arquivo existente.

8.2.3 Passo 3

Você pode evitar a destruição de um arquivo usando >> em vez de >. Ao usar >> você anexa a um arquivo. Execute os seguintes comandos para ver uma demonstração disso:

```
cat mymessage
echo "How are you?" >> mymessage
cat mymessage
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ cat mymessage
Greetings
sysadmin@localhost:~$ echo "How are you?" >> mymessage
sysadmin@localhost:~$ cat mymessage
Greetings
How are you?
sysadmin@localhost:~$
```

Observe que ao usar >> todos os dados existentes são preservados e os novos dados são anexados no final do arquivo.

8.2.4 Passo 4

O comando `find` é um bom comando para demonstrar como o `stderr` funciona. Este comando procura no sistema de arquivos por arquivos baseados em critérios como nome do arquivo. Execute o seguinte comando e observe a saída:

```
find /etc -name hosts
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ find /etc -name hosts
find: `/etc/ssl/private': Permission denied
/etc/hosts
sysadmin@localhost:~$
```

Observe a mensagem de erro indicando que você não tem permissão para acessar determinados arquivos / diretórios. Isso porque, como usuário comum, você não tem o direito de "olhar para dentro" de alguns diretórios. Esses tipos de mensagens de erro são enviados para o `stderr`, não para o `stdout`.

O comando `find` será abordado em maiores detalhes posteriormente. O comando está sendo usado agora para demonstrar a diferença entre `stdout` e `stderr`.

8.2.5 Passo 5

Para redirecionar o `stderr` (mensagens de erro) para um arquivo, emita o seguinte comando:

```
find /etc -name hosts 2> err.txt
cat err.txt
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ find /etc -name hosts 2> err.txt
/etc/hosts
sysadmin@localhost:~$ cat err.txt
find: `/etc/ssl/private': Permission denied
sysadmin@localhost:~$
```

Lembre-se de que o descritor de arquivo para `stderr` é o número 2, portanto, ele é usado junto com o símbolo `>` para redirecionar a saída `sdterr` para um arquivo chamado `err.txt`. Note que `1>` é o mesmo que `>`.

Nota: O exemplo anterior demonstra porque conhecer o redirecionamento é importante. Se você quiser "ignorar" os erros que o comando `find` exibe, você pode redirecionar essas mensagens para um arquivo e analisá-las mais tarde, facilitando o foco no restante da saída do comando.

8.2.6 Passo 6

Você também pode redirecionar `stdout` e `stderr` para dois arquivos separados.

```
find /etc -name hosts > std.out 2> std.err
cat std.err
cat std.out
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ find /etc -name hosts > std.out 2> std.err
sysadmin@localhost:~$ cat std.err
find: `/etc/ssl/private': Permission denied
sysadmin@localhost:~$ cat std.out
/etc/hosts
sysadmin@localhost:~$
```

Observe que um espaço é permitido, mas não é obrigatório após o símbolo de redirecionamento `>` .

8.2.7 Passo 7

Para redirecionar a saída padrão (`stdout`) e o erro padrão (`stderr`) para um arquivo, primeiro redirecione o `stdout` para um arquivo e, em seguida, redirecione o `stderr` para o mesmo arquivo usando a notação `2>&1`.

```
find /etc -name hosts >find.out 2>&1
cat find.out
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ find /etc -name hosts > find.out 2>&1
sysadmin@localhost:~$ cat find.out
find: `/etc/ssl/private': Permission denied
/etc/hosts
sysadmin@localhost:~$
```

A parte `2>&1` do comando significa enviar o `stderr` (canal 2) para o mesmo local onde o `stdout` (canal 1) está indo.

8.2.8 Passo 8

A entrada padrão (`stdin`) também pode ser redirecionada. Normalmente `stdin` vem do teclado, mas às vezes você quer que ele venha de um arquivo. Por exemplo, o comando `tr` traduz caracteres, mas só aceita dados de `stdin`, nunca de um nome de arquivo dado como argumento. Isso é ótimo quando você quer fazer algo como capitalizar dados que são introduzidos pelo teclado (Nota: Pressione **Control**+**d**, para sinalizar ao comando `tr` para parar o processamento da entrada padrão):

```
tr a-z A-Z
this is interesting
how do I stop this?
^D
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ tr a-z A-Z
this is interesting
THIS IS INTERESTING
how do I stop this?
HOW DO I STOP THIS?
sysadmin@localhost:~$
```

Nota: `^D` symboliza **Control**+**d**

8.2.9 Passo 9

O comando `tr` aceita entrada de teclado (`stdin`), traduz os caracteres e envia a saída para `stdout`. Para criar um arquivo com todos os caracteres minúsculos, execute o seguinte:

```
tr A-Z a-z > myfile

Wow, I SEE NOW

This WORKS!
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ tr A-Z a-z > myfile

Wow, I SEE NOW

This WORKS!

sysadmin@localhost:~$
```

Pressione a tecla **Enter** para se certificar de que seu cursor está na linha abaixo " This works!", Então use **Control+d** para parar a entrada. Para verificar se você criou o arquivo, execute o seguinte comando:

```
cat myfile
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ cat myfile

wow, i see now

this works!

sysadmin@localhost:~$
```

8.2.10 Passo 10

Execute os seguintes comandos para usar o comando `tr` redirecionando `stdin` de um arquivo:

```
cat myfile
tr a-z A-Z < myfile
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ cat myfile
wow, i see now
this works!
sysadmin@localhost:~$ tr a-z A-Z < myfile
WOW, I SEE NOW
THIS WORKS!
sysadmin@localhost:~$
```

8.2.11 Passo 11

Outra forma popular de redirecionamento é pegar a saída de um comando e enviá-lo para outro comando como entrada. Por exemplo, a saída de alguns comandos pode ser massiva, resultando na saída de rolagem da tela muito rapidamente para leitura. Execute o seguinte comando para obter a saída do comando `ls` e enviá-lo para o comando `more`, que exibe uma página de dados por vez:

```
ls -l /etc | more
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ ls -l /etc | more
total 372
-rw-r--r-- 1 root root    2981 Jan 28  2015 adduser.conf
-rw-r--r-- 1 root root      10 Jan 28  2015 adjtime
drwxr-xr-x 1 root root     900 Jan 29  2015 alternatives
drwxr-xr-x 1 root root     114 Jan 29  2015 apparmor.d
drwxr-xr-x 1 root root     168 Oct  1  2014 apt
-rw-r--r-- 1 root root    2076 Apr  3  2012 bash.bashrc
drwxr-xr-x 1 root root      72 Jan 28  2015 bash_completion.d
drwxr-sr-x 1 root bind     342 Jan 29  2015 bind
-rw-r--r-- 1 root root     356 Apr 19  2012 bindresvport.blacklist
-rw-r--r-- 1 root root     321 Mar 30  2012 blkid.conf
lrwxrwxrwx 1 root root      15 Jun 18  2014 blkid.tab -> /dev/.blkid.t
ab
drwxr-xr-x 1 root root      16 Jan 29  2015 ca-certificates
-rw-r--r-- 1 root root    7464 Jan 29  2015 ca-certificates.conf
drwxr-xr-x 1 root root      14 Jan 29  2015 calendar
drwxr-xr-x 1 root root      24 Jan 29  2015 cron.d
drwxr-xr-x 1 root root     134 Jan 29  2015 cron.daily
drwxr-xr-x 1 root root      24 Jan 29  2015 cron.hourly
drwxr-xr-x 1 root root      24 Jan 29  2015 cron.monthly
-rw-r--r-- 1 root root    2969 Mar 15  2012 debconf.conf
--More--
```

Você precisará pressionar a **barra de espaço** para continuar ou também pressionar **CTRL+c** para sair desta listagem.

O comando `cut` é útil para extrair campos de arquivos que são delimitados por um caractere, como os dois pontos (`:`) em `/etc/passwd` ou que possuem uma largura fixa. Ele será usado nos próximos exemplos, pois normalmente fornece uma grande quantidade de resultados que podemos usar para demonstrar o uso do caractere `|`.

8.2.12 Passo 12

No exemplo a seguir, você usará um comando chamado `cut` para extrair todos os nomes de usuários de um banco de dados chamado `/etc/passwd` (um arquivo que contém informações da conta do usuário). Primeiro, tente executar o comando `cut` por si só:

```
cut -d: -f1 /etc/passwd
```

Uma parte da saída do comando é mostrada no gráfico abaixo.

```
sysadmin@localhost:~$ cut -d: -f1 /etc/passwd
root
daemon
bin
sys
sync
games
man
lp
mail
news
uucp
proxy
www-data
backup
list
irc
gnats
nobody
libuuid
syslog
bind
sshd
operator
```

8.2.13 Passo 13

A saída no exemplo anterior foi desordenada e rolada para fora da tela. Na próxima etapa, você pegará a saída do comando `cut` e a enviará ao comando `sort` para fornecer alguma ordem à saída:

```
cut -d: -f1 /etc/passwd | sort
```

Uma parte da saída do comando é mostrada no gráfico abaixo.

```
sysadmin@localhost:~$ cut -d: -f1 /etc/passwd | sort
backup
bin
bind
daemon
games
gnats
irc
libuuid
list
lp
mail
man
news
nobody
operator
proxy
root
sshd
sync
sys
```

8.2.14 Passo 14

Agora a saída é classificada, mas ainda rola para fora da tela. Envie a saída do comando `sort` para o comando `more` para resolver este problema:

```
cut -d: -f1 /etc/passwd | sort | more
```

```
sysadmin@localhost:~$ cut -d: -f1 /etc/passwd | sort | more
backup
bin
bind
daemon
games
gnats
irc
libuuid
list
lp
mail
man
news
nobody
operator
proxy
root
sshd
sync
sys
sysadmin
syslog
uucp
--More--
```

8.3 Usando o find para encontrar arquivos

Nesta tarefa, você usará o comando `find` para localizar arquivos.

O comando `find` é um comando muito flexível com várias opções que permitem aos usuários localizar arquivos com base em uma ampla variedade de critérios, como nome do arquivo, tamanho, data, tipo e permissão. A construção básica do comando é:

```
find <directory to start search> -criteria <argument for criteria>
```

O comando iniciará a pesquisa no diretório especificado e pesquisará recursivamente todos os subdiretórios.

8.3.1 Passo 1

Procure por arquivos que iniciem no seu diretório pessoal contendo o nome `bash`.

```
find ~ -name "*bash*"
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ find ~ -name "*bash*"
/home/sysadmin/.bash_logout
/home/sysadmin/.bashrc
sysadmin@localhost:~$
```

Lembre-se de que `~` é usado para representar seu diretório pessoal.

8.3.2 Passo 2

Encontre arquivos que foram modificados (ou criados) há menos de 5 minutos no diretório especificado usando os seguintes comandos:

```
find ~/Music -mmin -5
touch ~/Music/mysong
find ~/Music -mmin -5
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ find ~/Music -mmin -5
sysadmin@localhost:~$ touch ~/Music/mysong
sysadmin@localhost:~$ find ~/Music -mmin -5
/home/sysadmin/Music
/home/sysadmin/Music/mysong
sysadmin@localhost:~$
```

O primeiro comando `find` não encontra arquivos que foram modificados dentro do tempo especificado. Em seguida, você criou um arquivo usando o comando `touch` e executou o comando `find` novamente, resultando no comando `find` descobrindo o novo arquivo.

O diretório `Music` foi exibido com o segundo comando `find` porque o diretório foi modificado, o resultado de um arquivo sendo adicionado ao diretório.

8.3.3 Passo 3

Execute o seguinte comando para encontrar arquivos no diretório `/usr` que tenham mais de 2 MB de tamanho:

```
find /usr -size +2M
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ find /usr -size +2M
/usr/bin/python2.7
/usr/lib/perl/5.14.2/auto/Encode/JP/JP.so
/usr/lib/perl/5.14.2/auto/Encode/KR/KR.so
/usr/share/GeoIP/GeoIPv6.dat
/usr/share/file/magic.mgc
sysadmin@localhost:~$
```

8.3.4 Passo 4

Encontre arquivos do tipo “diretório” no local especificado.

```
find /usr/share/bug -type d
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ find /usr/share/bug -type d
/usr/share/bug
/usr/share/bug/apt
/usr/share/bug/gnupg
/usr/share/bug/initramfs-tools
/usr/share/bug/procps
/usr/share/bug/cron
/usr/share/bug/file
/usr/share/bug/libmagic1
/usr/share/bug/logrotate
/usr/share/bug/man-db
/usr/share/bug/vim-tiny
sysadmin@localhost:~$
```

8.3.5 Passo 5

Para verificar se a saída exibe diretórios, utilize a opção `-ls`. O comando `find` usa a opção `-print` por padrão, que exibe apenas nomes de arquivos. A opção `-ls` fornece detalhes do arquivo:

```
find /usr/share/bug -type d -ls
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ find /usr/share/bug -type d -ls
  2280    0 drwxr-xr-x   1 root     root          138 Jan 29  2015 /usr/share/bug
  2281    0 drwxr-xr-x   1 root     root           12 Jan 28  2015 /usr/share/bu
g/apt
  2283    0 drwxr-xr-x   1 root     root           14 Jan 28  2015 /usr/share/bug/gnupg
  2285    0 drwxr-xr-x   1 root     root           12 Jan 28  2015 /usr/share/bug/initramfs-tools
  2287    0 drwxr-xr-x   1 root     root           14 Jan 28  2015 /usr/share/bug/procps
 10784    0 drwxr-xr-x   1 root     root           26 Jan 29  2015 /usr/share/bug/cron
 10787    0 drwxr-xr-x   1 root     root           28 Jan 29  2015 /usr/share/bug/file
 10790    0 drwxr-xr-x   1 root     root           28 Jan 29  2015 /usr/share/bug/libmagic1
 10793    0 drwxr-xr-x   1 root     root           12 Jan 29  2015 /usr/share/bug/logrotate
10795    0 drwxr-xr-x   1 root     root           14 Jan 29  2015 /usr/share/bu
g/man-db
 10797    0 drwxr-xr-x   1 root     root           26 Jan 29  2015 /usr/share/bug/vim-tiny
sysadmin@localhost:~$
```

Lembre-se de que o caractere `d` antes das permissões `rwxr-x-r-x` indica que o arquivo é, na verdade, um diretório.

8.4 Exibindo arquivos de texto grandes

Embora arquivos de texto grandes possam ser exibidos com o comando `cat`, é inconveniente rolar para trás na direção do topo do arquivo. Além disso, arquivos realmente grandes não podem ser exibidos dessa maneira porque a janela do terminal armazena apenas um número específico de linhas de saída na memória.

O uso dos comandos `more` ou `less` permite que o usuário visualize uma "página" ou uma linha por vez. Esses comandos "pager" também permitem outras formas de navegação e busca que serão demonstradas nesta seção.

Nota: os exemplos são dados usando os comandos `more` e `less`. Na maioria das vezes, os comandos funcionam da mesma forma, porém o comando `less` é mais avançado e tem mais recursos. O comando `more` é ainda importante saber, porque algumas distribuições do Linux não têm o comando `less`, mas todas as distribuições do Linux têm o comando `more`.

Se você não estiver interessado em visualizar o arquivo inteiro ou a saída como um comando, existem vários comandos que podem filtrar o conteúdo do arquivo ou da saída. Nesta seção, você aprenderá o uso dos comandos `head` e `tail` para extrair informações da parte superior ou inferior da saída de um comando ou conteúdo do arquivo.

8.4.1 Passo 1

O `/etc/passwd` provavelmente é muito grande para ser exibido na tela sem rolar a tela. Para ver uma demonstração disso, use o comando `cat` para exibir todo o conteúdo do arquivo `/etc/passwd`:

```
cat /etc/passwd
```

Sua saída deve ser semelhante à seguinte:

```
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/s
h
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
bind:x:102:105::/var/cache/bind:/bin/false
sshd:x:103:65534::/var/run/sshd:/usr/sbin/nologin
operator:x:1000:37::/root:/bin/sh
sysadmin:x:1001:1001:System Administrator,,,,:/home/sysadmin:/bin/bash
sysadmin@localhost:~$
```

8.4.2 Passo 2

Use o comando `more` para exibir todo o conteúdo do arquivo `/etc/passwd`:

```
more  /etc/passwd
```

Sua saída deve ser semelhante à seguinte:

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/s
h
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
bind:x:102:105::/var/cache/bind:/bin/false
sshd:x:103:65534::/var/run/sshd:/usr/sbin/nologin
operator:x:1000:37::/root:/bin/sh
--More--(92%)
```

Nota: O `--More--(92%)` indica que você está "in" no comando `more` e 92% nos dados atuais.

8.4.3 Passo 3

Enquanto estiver no comando `more`, você pode visualizar a tela de ajuda pressionando a tecla **h**:

```
h
```

Sua saída deve ser semelhante à seguinte:

```
Most commands optionally preceded by integer argument k.  Defaults in
brackets

Star (*) indicates argument becomes new default.

-------------------------------------------------------------------------
--------

<space>                 Display next k lines of text [current screen s
ize]

z                       Display next k lines of text [current screen s
ize]*

<return>                Display next k lines of text [1]*

d or ctrl-D             Scroll k lines [current scroll size, initially
11]*

q or Q or <interrupt>   Exit from more

s                       Skip forward k lines of text [1]

f                       Skip forward k screenfuls of text [1]

b or ctrl-B             Skip backwards k screenfuls of text [1]

'                       Go to place where previous search started

=                       Display current line number

/<regular expression>   Search for kth occurrence of regular expressio
n [1]

n                       Search for kth occurrence of last r.e [1]

!<cmd> or :!<cmd>       Execute <cmd> in a subshell

v                       Start up /usr/bin/vi at current line

ctrl-L                  Redraw screen

:n                      Go to kth next file [1]

:p                      Go to kth previous file [1]

:f                      Display current file name and line number

.                       Repeat previous command

-------------------------------------------------------------------------
--------

--More--(92%)
```

8.4.4 Passo 4

Pressione a **barra de espaço** para ver o restante do documento:

```
<SPACE>
```

No próximo exemplo, você aprenderá a pesquisar um documento usando os comandos `more` ou `less`.

Procurar por um padrão dentro dos comandos `more` e `less` é feito digitando a barra */*, seguida do padrão para encontrar. Se uma correspondência for encontrada, a tela deverá rolar para a primeira correspondência. Para avançar para a próxima correspondência, pressione a tecla **n** . Com o comando `less`, você também pode voltar para as correspondências anteriores pressionando a tecla **N** (n maiúsculo).

8.4.5 Passo 5

Use o comando `less` para exibir todo o conteúdo do arquivo `/etc/passwd`. Em seguida, procure a palavra `bin`, use **n** para avançar e **N** para retroceder. Finalmente, saia do pager `less`, digitando a letra **q**:

```
less /etc/passwd

/bin

nnnNNNq
```

Importante: Ao contrário do comando `more`, que sai automaticamente quando você atinge o final de um arquivo, você deve pressionar uma tecla de saída, como **q**, para sair do programa `less`.

8.4.6 Passo 6

Você pode usar o comando `head` para exibir a parte superior de um arquivo. Por padrão, o comando `head` exibirá as dez primeiras linhas do arquivo:

```
head /etc/passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ head /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
sysadmin@localhost:~$
```

8.4.7 Passo 7

Use o comando `tail` para exibir as últimas dez linhas do arquivo `/etc/passwd`:

```
tail /etc/passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ tail /etc/passwd
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/s
h
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
bind:x:102:105::/var/cache/bind:/bin/false
sshd:x:103:65534::/var/run/sshd:/usr/sbin/nologin
operator:x:1000:37::/root:/bin/sh
sysadmin:x:1001:1001:System Administrator,,,,:/home/sysadmin:/bin/bash
sysadmin@localhost:~$
```

8.4.8 Passo 8

Use o comando `head` para exibir as duas primeiras linhas do arquivo `/etc/passwd`:

```
head -2 /etc/passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ head -2 /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
sysadmin@localhost:~$
```

8.4.9 Passo 9

Execute a seguinte linha de comando para enviar a saída do comando `ls` para o comando `tail`, exibindo os últimos cinco nomes de arquivos no diretório `/etc`:

```
ls /etc | tail -5
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$  ls /etc | tail -5
update-motd.d
updatedb.conf
vim
wgetrc
xml
sysadmin@localhost:~$
```

Como você viu, os comandos `head` e `tail` emitem dez linhas por padrão. Você também pode usar uma opção `-#` (ou você pode usar a opção `-n #`, onde `#` é um número de linhas para a saída). Ambos os comandos podem ser usados para ler a entrada padrão de um pipe que recebe a saída de um comando.

Você também viu onde os comandos `head` e `tail` são diferentes: o comando `head` começa a contar suas linhas para a saída da parte superior dos dados, enquanto o comando `tail` conta o número de linhas para a saída da parte inferior dos dados. Existem algumas diferenças adicionais entre esses dois comandos, conforme demonstrado nas próximas tarefas.

8.4.10 Passo 10

Outra maneira de especificar quantas linhas para a saída com o comando `head` é usar a opção `-n -#`, onde `#` é o número de linhas contadas a partir da parte inferior da saída a ser excluída. Observe o símbolo menos `-` na frente do `#`. Por exemplo, se o `/etc/passwd` contiver 24 linhas e o seguinte comando exibir as linhas 1-4, excluindo as últimas vinte linhas:

```
head -n -20 /etc/passwd
```

```
sysadmin@localhost:~$ head -n -20 /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sysadmin@localhost:~$
```

8.5 Pesquisando texto usando expressões regulares

Nesta tarefa, você usará a família `grep` de comandos com expressões regulares para procurar uma cadeia de caracteres específica em um fluxo de dados (por exemplo, um arquivo de texto).

O comando `grep` usa expressões regulares básicas, caracteres especiais como curingas que correspondem aos padrões nos dados. O comando `grep` retorna a linha inteira contendo o padrão correspondente.

A opção `-E` para o comando grep pode ser usada para realizar pesquisas com expressões regulares estendidas, essencialmente expressões regulares mais poderosas. Outra maneira de usar expressões regulares estendidas é usar o comando `egrep`.

O comando `fgrep` é usado para corresponder caracteres literais, ignorando o significado especial de caracteres de expressão regular.

8.5.1 Passo 1

O uso do `grep` em sua forma mais simples é procurar por uma determinada string de caracteres, como o `sshd` no arquivo `/etc/passwd`. O comando `grep` irá imprimir a linha inteira contendo a correspondência:

```
cd /etc

grep sshd passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:~$ cd /etc

sysadmin@localhost:/etc$ grep sshd passwd

sshd:x:103:65534::/var/run/sshd:/usr/sbin/nologin

sysadmin@localhost:/etc$
```

8.5.2 Passo 2

Expressões regulares são "gananciosas" no sentido de que elas corresponderão a todas as instâncias do padrão especificado:

```
grep root passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:/etc$ grep root passwd
root:x:0:0:root:/root:/bin/bash
operator:x:1000:37::/root:/bin/sh
sysadmin@localhost:/etc$
```

Observe os destaques vermelhos indicam exatamente o que foi correspondido. Você também pode ver que todas as ocorrências de `root` foram correspondidas em cada linha.

8.5.3 Passo 3

Para limitar a saída, você pode usar expressões regulares para especificar um padrão mais preciso. Por exemplo, o caractere circunflexo (`^`) pode ser usado para corresponder a um padrão no início de uma linha; Portanto, quando você executa a seguinte linha de comando, apenas as linhas que começam com `root` devem ser correspondidas e exibidas:

```
grep '^root' passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:/etc$ grep '^root' passwd
root:x:0:0:root:/root:/bin/bash
sysadmin@localhost:/etc$
```

Observe que há duas instâncias adicionais da palavra `root`, mas apenas a que aparece no início da linha é correspondida (exibida em vermelho).

Melhor prática: Use aspas simples (não aspas duplas) em torno de expressões regulares para evitar que o programa de shell tente interpretá-las.

8.5.4 Passo 4

Corresponda o padrão `sync` em qualquer lugar de uma linha:

```
grep 'sync' passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:/etc$ grep 'sync' passwd
sync:x:4:65534:sync:/bin:/bin/sync
sysadmin@localhost:/etc$
```

8.5.5 Passo 5

Use o símbolo `$` para corresponder ao padrão `sync` no final de uma linha:

```
grep 'sync$' passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:/etc$ grep 'sync$' passwd
sync:x:4:65534:sync:/bin:/bin/sync
sysadmin@localhost:/etc$
```

O primeiro comando corresponde a todas as instâncias; o segundo corresponde apenas à instância no final da linha.

8.5.6 Passo 6

Use o caractere de período. para combinar com qualquer caractere único. Por exemplo, execute o seguinte comando para corresponder a qualquer caractere seguido por um '`y`':

```
grep '.y' passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:/etc$ grep '.y' passwd
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
proxy:x:13:13:proxy:/bin:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
sysadmin:x:1001:1001:System Administrator,,,,:/home/sysadmin:/bin/bash
sysadmin@localhost:/etc$
```

8.5.7 Passo 7

O caractere pipe, `|` ou "operador de alternação" atua como um operador "ou". Por exemplo, execute o seguinte para tentar corresponder a `sshd`, `root` ou `operator`:

```
grep 'sshd|root|operator' passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:/etc$ grep 'sshd|root|operator' passwd
sysadmin@localhost:/etc$
```

Observe que o comando `grep` não reconhece o pipe como o operador de alternação por padrão. O comando `grep` está realmente incluindo o pipe como um caractere simples no padrão a ser correspondido. O uso de `grep -E` ou `egrep` permitirá o uso de expressões regulares estendidas, incluindo alternância.

8.5.8 Passo 8

Use o comutador `-E` para permitir que o `grep` opere no modo estendido para reconhecer o operador de alternância:

```
grep -E 'sshd|root|operator' passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:/etc$ grep -E 'sshd|root|operator' passwd
root:x:0:0:root:/root:/bin/bash
sshd:x:103:65534::/var/run/sshd:/usr/sbin/nologin
operator:x:1000:37::/root:/bin/sh
sysadmin@localhost:/etc$
```

8.5.9 Passo 9

Use outra expressão regular estendida, desta vez com `egrep` com alternância em um grupo para corresponder a um padrão. As strings `nob` e `non` irão corresponder:

```
egrep 'no(b|n)' passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:/etc$ egrep 'no(b|n)' passwd
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
sysadmin@localhost:/etc$
```

Nota: Os parênteses, `( )`, foram usados para limitar o "escopo" do caractere `|`. Sem eles, com `nob|n`, o padrão significaria "combinar `nob` ou `n`"

8.5.10 Passo 10

Os caracteres `[ ]` também podem ser usados para corresponder a um único caractere, no entanto, ao contrário do caractere ponto `.`, Os caracteres `[ ]` são usados para especificar exatamente qual caractere você deseja corresponder. Por exemplo, se você quiser corresponder a um caractere numérico, poderá especificar `[0-9]`. Execute o seguinte comando para uma demonstração:

```
head passwd | grep '[0-9]'
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:/etc$ head passwd | grep '[0-9]'
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
sysadmin@localhost:/etc$
```

Nota: O comando `head` foi usado para limitar a saída do comando `grep`.

8.5.11 Passo 11

Suponha que você queira procurar um padrão que contenha uma sequência de três dígitos. Você pode usar `{ }` caracteres com um número para expressar que deseja repetir um padrão um número específico de vezes; por exemplo: `{3}` O uso do qualificador numérico requer o modo estendido do `grep`:

```
grep -E '[0-9]{3}' passwd
```

Sua saída deve ser semelhante à seguinte:

```
sysadmin@localhost:/etc$ grep -E '[0-9]{3}' passwd
sync:x:4:65534:sync:/bin:/bin/sync
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
bind:x:102:105::/var/cache/bind:/bin/false
sshd:x:103:65534::/var/run/sshd:/usr/sbin/nologin
operator:x:1000:37::/root:/bin/sh
sysadmin:x:1001:1001:System Administrator,,,,:/home/sysadmin:/bin/bash
sysadmin@localhost:/etc$
```