Curso de Formação LPIC-1

Exame 101




Bruno Santos <brunosantos@whatgeek.com.pt>

Curso Linux: formação

- Trabalhar na linha de comandos
  - Papel da Shell
  - Shells
  - Identificar a shell
  - Alterar a shell
  - sh: Prompts
  - bash: Bourne Again Shell
  - bash: Comand Editing
  - bash: Comand Completation
  - Shell/Variáveis de ambiente
  - Variáveis de ambiente chave

Processamento de texto

Papel de uma Shell

- Programa para interagir com o sistema
- Interface de texto entre o Kernel e o Utilizador
- Uma prompt, representada na forma mais simples por # ou $
- *Bash* usualmente por defeito
- Especificada em /etc/passwd por utilizador

	Ash	**bash**	**C-shell**	**PD-ksh**	**T-shell**	**Zsh**
Binário	ash	bash	csh	pdksh	tcsh	zsh
Controlo de processos	N	S	S	S	S	S
Aliases	N	S	S	S	S	S
Funções	Y	S	N	S	N	S
Redireccionamentos	S	S	S	S	S	S
Histórico	N	S	S	S	S	S
Edição	N	S	N	S	S	S
Completação	N	S	S	S	S	S

Linha de comandos

Shell: Login

- Uma *Shell* de *login* é executada quando se entra no sistema
- O ficheiro */etc/profile* é lido (sourced). Este é o ficheiro de configuração global se for usada a *bash*
- Uma sessão de utilizador:
  - O utilizador autentica-se com *username/password*
  - Lido */etc/profile*
  - Lido *~/.bash_profile*
  - Lido *~/.bashrc* desde o *~/.bash_profile*
  - Utilizador trabalha na sua sessão
  - Utilizador sai da sessão com *logout*, *exit* ou *Ctrl+D*
  - Lido *~/.bash_logout*

Linha de comandos

Shell: Login

Ordem da execução dos ficheiros de configuração		
Globais	***/etc/profile***	***/etc/bashrc***
Locais	***~/.bash_profile***	
	~/.bashrc	
	~/.profile (opcional)	
	~/.bash_login (opcional)	
	~/.bash_logout	

- */etc/profile*
- *~/.bash_profile* (se existir) (se existirem os 3, é lido apenas este)
  - *~/.bash_login*
  - *~/.profile*
- Contém variáveis, código e definições que afectam apenas o utilizador em questão
- Chama *~/.bashrc*

Linha de comandos

Identificar a shell em uso

- Em Linux, a liberdade é escolher.
- Com a existência de várias shells, como saber com qual trabalhamos?

```
echo $SHELL
/bin/bash
```

```
cat /etc/passwd | grep -i $USER
feiticeir0:x:1000:1002::/home/feiticeir0:/bin/bash
```

Linha de comandos

Shell: Comandos

- *Comando opções argumentos*

```
ls -R /etc/profiles
```

- Separar comandos longos
- Uso do caracter \

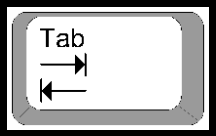
```
rpm -q1 pacote-1.1.rpm | xargs ficheiro | grep  -i LSB | nl | pr | tac
```

```
rpm -q1 pacote-1.1.rpm \
xargs ficheiro \
grep  -i LSB \
nl \
pr \
tac
```

Linha de comandos

Shell: Comandos

- Completação de comandos
- Tecla <TAB>




```
/bin/ls <TAB>
ls      lsattr  lsmod
feiticeir0@prometheus ~ $ /bin/ls
```

```
cd Do
Documents/ Downloads/
feiticeir0@prometheus ~ $ cd Documents/Fo
Formacao_LPI/ Fotografias/
feiticeir0@prometheus ~ $ cd Documents/Formacao_LPI/Aula0
Aula01/    Aula01.odp  Aula02/    Aula03/    Aula04/    Aula05/
feiticeir0@prometheus ~ $ cd Documents/Formacao_LPI/Aula0
```

Linha de comandos

Shell: Caracteres especiais

Caracteres especiais mais comuns		
Caracter	**Uso**	**Exemplo**
~	Atalho para pasta pessoal do utilizador	**vi ~/.bashrc**
\	Ignorar o caracter seguinte (caracter esc)	**echo $PRD is \$5**
/	Separador de directorias	**cd /home/bruno**
$	Variável – Antecede qualquer variável	**echo $VAR**
?	Meta-caracter único	**ls *.t?t**
'	Apóstrofo - Strings	**echo 'Custo: $100'**
`	Acento grave - substituição	**echo `date`**
“	Aspas - Strings	**echo “Custo: $VAR”**

Linha de comandos

Shell: Caracteres especiais

Caracteres especiais mais comuns (continuação)		
Caracter	**Uso**	**Exemplo**
&	Executar um trabalho em plano de fundo	**firefox &**
&&	Se cmd1 terminar com 0 (sucesso) executar cmd2	**cmd1 && cmd2**
|	Envia a saída de um comando como entrada de outro (pipe)	**ls -l | pr**
||	Se cmd1 falhar, executar cmd2	**cmd1 || cmd2**
;	Executa vários comandos	**cmd1; cmd2**
[]	Intervalo de letras/números	**ls ficheiro[0-9]**
>	Redirecciona a saída para um ficheiro	**prog1 > ficheiro**
<	Redirecciona a entrada para um comando	**prog1 < ficheiro**

Linha de comandos

Shell: Controlar execução de comandos

- Na execução de um comando, existe sempre um estado de saída
- Não é mostrado, a não ser que uma mensagem seja enviada para a saída standard

```
ls -a ; echo $?
.  ..  install-amd64-minimal-20090521.iso  install-amd64-minimal-20090604.iso  LXF  Torrent
0
```

- Estado 0 : comando executado com sucesso
- Estado 127 ou 1 (não zero) : comando executado sem sucesso

```
foo ; echo $?
bash: foo: command not found
127
```

Linha de comandos

Shell: Executar múltiplos comandos

- Múltiplas formas de executar vários comandos com apenas um <ENTER>
- Caracteres especiais ou comandos múltiplos
  - ; - todos os comandos são executados independentemente
  - && - o segundo comando só é executado se o primeiro executar com sucesso
  - || - o segundo comando só é executado se o primeiro falhar

```
make modules ; echo DO MAKE MODULES_INSTALL NEXT
```

```
make && make modules_install
```

```
tar -czvf /dev/st0  / || mail root - “doh, backup failed”
```

Linha de comandos

Shell: Substituição de comandos

- Necessidade de “agarrar” na saída de um comando e colocar numa variável
- Uso do ` (apóstrofo) ou agrupar o comando com $(comando)
- Comando *export* é usado para exportar uma variável

```
export DATA=`date`
feiticeir0@prometheus ~ $ echo $DATA
Tue Jun 16 22:44:45 WEST 2009
```

```
echo "O meu kernel é $(uname -s ) na release $(uname -r) numa arquitectura $(uname -m)”
O meu kernel é Linux na release 2.6.29-zen1 numa arquitectura x86_64
```

Linha de comandos

Shell: Histórico

- O comando *history* mostra uma listagem dos comandos já executados
- Variável HISTFILE
  - Geralmente definida como ~/.bash_history
  - echo $HISTFILE
- Quando um utilizador entra, o ficheiro é aberto.
- Normalmente, não é escrito até que o utilizador termine a sessão
- No caso de uma sub-shell, esta lê o ficheiro da shell parente e escreve no ficheiro no final

```
echo “Esta é a shell principal”
bash
history
echo “Isto é o histórico de uma sub-shell”
history
exit
history
tail $HISTFILE
```

Linha de comandos

Shell: Histórico – Variáveis Importantes

- HISTFILE – Por defeito em ~/.bash_history e é definido no ambiente aquando do login
- HISTCMD – O histórico ou o número de indexação do comando corrente. echo $HISTCMD mostra o número para o comando
- HISTCONTROL
  - Se definido para *ignorespace,* linhas que terminem num espaço não são adicionadas ao histórico.
  - Se definido para *ignoredups*, linhas que duplicam a linha anterior são ignoradas
- HISTFILESIZE – O número de linhas usadas para o histórico aquando da sua escrita no ficheiro. Se o resultado for superior que o indicado, é truncado a partir do inicio para corresponder ao valor correcto. Por defeito é 500

Linha de comandos

Shell: Histórico - fc

- O comando *fc* lista partes da pilha do histórico ou edita uma linha, ou várias linhas no editor por defeito

```
fc
```

- Editar uma série de linhas
  - fc x y
- Listar uma série de linhas
  - fc -l x y

```
fc 78 85
fc -l 78 85
```

Linha de comandos

Shell: Variáveis de ambiente

- O processo pai de todos os processos numa máquina Linux é o processo *init*, com um PID (*process ID)* de 1.
- O binário do processo *init* é */etc/init*, e o seu ambiente é propagado por todos os processos filhos.
- Definido no *init* estão uma série de caminhos (*paths)* que são o básico para todos os caminhos adicionados pelos ficheiros de ambiente. O caminho por defeito ou a base de todos os caminhos é: */usr/local/sbin:/sbin:/usr/sbin:/usr/bin*
- Visualizar as variáveis de ambiente do sistema no seu estado puro é dificil, pois para as ver requer que o utilizador entre na sua sessão e execute os mesmos *scripts* que está a visualizar.

```
env
```

Linha de comandos

Shell: path

- Uma listagem de directorias separadas por : (dois pontos) onde são efectuadas as procuras por ficheiros executáveis.
- É normalmente usada para procurar comandos que não se encontram na directoria corrente, excepto se a directoria estiver na *path.*
- Pode-se sempre referir a um executável na directoria corrente de duas formas:
  - Caminho absoluto: /home/feiticeir0/comando
  - Caminho relativo: ; ./comando

```
mkdir bin
cd bin
echo “ls -l” >> ls2
Chmod +x ls2
/home/$USER/bin/ls
./ls
```

Linha de comandos

Shell: path

- Definida geralmente em */etc/profile* para todos os utilizadores
- ~/.bash_profile para cada utilizador
- Adicionar um novo caminho sem alterar os já existentes é muito simples

```
nano .bash_profile
<navegar até ao final do ficheiro>
Adicionar a seguinte linha:
export PATH=$PATH:~/bin
```

Linha de comandos

Shell: $HOME

- Alguns atalhos e variáveis apontam directamente para a home de um utilizador
- A variável *HOME* é lida da entrada do utilizador do ficheiro */etc/passwd*
- O valor da variável é o caminho absoluto para a directoria do utilizador

```
cd $HOME
cd ~
cd
```

Linha de comandos

Shell: Prompts

- Quatro possiveis prompts existem num sistema Linux: PS1 até PS4
- Um utilizador normalmente só vê uma: PS1 (# ou $)
- Se ele comete um erro tipográfico ou de sintax, pode ser confrontado com duas sitações:
  - A continuação
  - Ou a prompt PS2 (>)
- Em Red Hat, PS1 é definido em /etc/bashrc
- Em Debian, PS1 é definido em /etc/bashrc e /etc/profile
- A variável PS1 é definida pela *prompt* do utilizador, que é mostrada cada vez que o utilizador inicia a sua sessão
- PS2 normalmente apenas mostra linhas quando:
  - são continuadas, como quebrar a linha de comandos com \
  - Um parêntese, apostrofo ou aspas são deixadas abertas
  - PS3 e PS4 raramente são utilizadas e podem não existir

Linha de comandos

Shell: Prompts

Código	Mostra
\a	Caracter ASCII para a campainha (07)
\d	A data em dia da semana, mês e dia (ex: Seg Maio 26)
\e	ASCII caracter de escape (033)
\h	
\H	
\j	
\l	
\n	
\r	
\s	
\t	
\T	
\@	
\A	

Linha de comandos

Shell: Prompts

Código	Mostra
\u	O nome de utilizador do utilizador corrente
\v	A versão da bash (2.00))
\V	A *release* da bash, versão mais minor release (2.00.0)
\w	A directoria corrente
\W	O nome base da directoria corrente
\!	O número na pilha do histórico do comando corrente
\#	O número de comando do comando corrente
\$	Se o UID é 0, é #, senão é $
\nnn	O caracter correspondente ao valor octal de nnn
\\	contra-barra
\[	Começa uma sequência de caracteres não imprimiveis, que podem ser usados para embeber uma sequência de controlo na bash
\]	Termina uma sequência de caracteres não imprimiveis

Linha de comandos

Shell: Prompts

```
nano -w .bashrc
PS1=”[\u@\h local \W]\\$”
Ctrl + o para sair
source .bashrc
```

- Nome do computador + @ + um beep
- Nome da consola + OO + nova linha
- Tempo em 24h (HH:MM) + ? + directoria corrente + numero de processos

Linha de comandos

Shell: biblioteca Readline

- Fornece a capacidades de edição de texto na bash
- Mapa de teclas do editor emacs
- Ficheiro de configuração ~/.inputrc

```
# Set various bindings for emacs mode.

set editing-mode emacs

$if mode=emacs

Meta-Control-h:       backward-kill-word   Text after the
function name is ignored

#
# Arrow keys in keypad mode
#
#"\M-OD":        backward-char
#"\M-OC":        forward-char
#"\M-OA":        previous-history
```

Linha de comandos

Shell: biblioteca Readline

- Algumas opções
  - set editing-mode=vi
  - bell-style (audible | visible)
  - enable-keypad (on | off)
  - mMark-directories (on | off)
- Alguns comandos na shell
  - Ctrl + f – move um caracter
  - Ctrl + a – move-se para o inicio de uma linha
  - Ctrl + l – limpa o ecrâ

Linha de comandos

Shell: Opções na bash

- Comando *set*
- Activa ou desactiva opções da bash
  - set -o opção (activa a opção)
  - set +o opção (desactiva a opção)
- Opções mais comuns:
  - emacs ou vi – define o mapa de teclas para editar na linha de comandos
  - history – opção activada por defeito. O valor da variável HISTFILE é lido para determinar o ficheiro do histórico
  - hashall – activado por defeito. Activa uma tabela de hash dos comandos requeridos e das suas localizações para uso repetido do comando
  - monitor – Esta opção obriga o controlo de trabalhos a executar os processos em plano de fundo num grupo separado e a notificar a consola quando estes terminem por completo
  - noclobber – Esta opção está desactivada. Quando activada, NÃO PERMITE reescrever um ficheiro por um simbolo de redireccionamento (>). Um erro de sintaxe ocorre se isto acontecer. O uso de duplo redireccionamento (>>) é aconselhado

Linha de comandos

Shell: Opções na bash

- Opções mais comuns (continuação):
  - noexec – Quando activada, corre os scripts, mas não afecta em nada o sistema.
  - notify – Reporta trabalhos completos para a consola imediatamente, em vez de esperar pela próxima execução do comando *jobs*
  - verbose – Esta opção mostra no terminal quaisquer comandos antes de serem executados.

Curso Linux

bibliografia

- LPIC I, Exam Cram 2, Brunson - QUE Certification
- LPI Linux Certification In a Nutshell, Pritchard, Pessanha, Langfeldt, Stranger & Dean – O REILLY
- Linux Administration Handbook, Second edition, Nemeth Snyder Hein – Prentice Hall