

Resumo Importante sobre o Curso de LPIC-1

Tópico 103.1 - Trabalhando na Linha de Comando – Comando which e Quoting

Douglas Luna - @DipionX02  

O que estudamos nesta aula?

O comando which	2
Quoting	3
Aspas Simples - Single Quotes	3
Aspas Duplas - Double Quotes	4
Barra Invertida - Escape Character	5

O comando which

O comando which tem ligação com o PATH, local onde o bash vai procurar os diretórios de um binário, o which é uma ferramenta de busca serve para procurar localizar um comando, ele vai nos diretórios configurados na variável de ambiente PATH e encontrado aquele valor que foi passado como parâmetro para ele. Por exemplo queremos localizar o comando echo, executamos:

```
$ which echo
```

```
kali@kali:~$ which echo
/usr/bin/echo
kali@kali:~$
```

Ele pode localizar qualquer arquivo que estiver dentro desses diretórios do PATH. Em breve veremos mais comandos de localização.

Outro exemplo:

```
kali@kali:~$ which nmap
/usr/bin/nmap
kali@kali:~$ which python
/usr/bin/python
kali@kali:~$
```

Quoting

No bash temos uma série de caracteres especiais, por exemplo o “\$” que usamos com a variável, como no comando “echo \$PATH” para obtermos os diretórios do bash.

A primeira coisa que o shell faz e interpretar as variáveis e depois ele executa o comando com o resultado dessa interpretação de variáveis.

As aspas duplas, aspas simples e a barra invertida existe justamente para proteger, impedir e controlar a interpretação dessas variáveis.

Se executarmos o comando “**echo ***” a primeira coisa que o shell fará e interpretar o asterisco, o asterisco significa todos os arquivos do diretório local. Ao executar o comando veremos todos os arquivos, conforme abaixo:

```
$ echo *
```

```
kali@kali:~/LPIC-1$ echo *
103.1 Trabalhando na Linha de Comando - ComandosSequenciais, history, man.docx Aula 103.1 - Execucao de comandos e
m sequencia Aula 103.1 - Trabalhando na linha de comando - uname, alias.pdf original.tgz
kali@kali:~/LPIC-1$
```

Aspas Simples - Single Quotes

Incluir caracteres entre aspas simples (‘’) preserva o valor literal de cada caractere entre aspas. Uma aspas simples pode não ocorrer entre aspas simples, mesmo quando precedida por uma barra invertida. Exemplo:

```
$ echo ‘*’
```

```
kali@kali:~/LPIC-1$ echo ‘*’
*
kali@kali:~/LPIC-1$
```

Exemplo, criação a variável TESTE=Curso e vamos usar as aspas simples para proteger:

```
kali@kali:~/LPIC-1$ TESTE=Curso
kali@kali:~/LPIC-1$ echo $TESTE
Curso
kali@kali:~/LPIC-1$ echo ‘$TESTE’
$TESTE
kali@kali:~/LPIC-1$
```

Note que o CIFRÃO FOI IGNORADO e não temos o retorno da variável TESTE.

Aspas Duplas - Double Quotes

A inclusão de caracteres entre aspas duplas (‘ “ ‘) preserva o valor literal de todos os caracteres entre aspas, com exceção de:

'\$', - Cifrão

'`', - Crase

'\'', - Barra normal

e, quando a expansão do histórico estiver ativada, '!'. o shell está no modo POSIX (consulte Modo Bash POSIX), o '!' não tem significado especial entre aspas duplas, mesmo quando a expansão do histórico está ativada. Os caracteres '\$' e '"' mantêm seu significado especial entre aspas duplas (consulte Expansões do Shell). A barra invertida mantém seu significado especial somente quando seguido por um dos seguintes caracteres: '\$', '"', "'", '\', ou nova linha. Nas aspas duplas, as barras invertidas que são seguidas por um desses caracteres são removidas. As barras invertidas que precedem os caracteres sem um significado especial não são modificadas. Uma citação dupla pode ser citada entre aspas duplas precedendo-a com uma barra invertida. Se ativada, a expansão do histórico será realizada, a menos que um '!' Que apareça entre aspas duplas seja escapado usando uma barra invertida. A barra invertida que precede o '!' Não é removida.

Os parâmetros especiais '*' e '@' têm um significado especial entre aspas duplas (consulte Expansão dos parâmetros do shell).

Se eu fizer o comando `echo *` com o asterisco protegido por aspas duplas, o shell impede a execução do asterisco e imprime o asterisco conforme o comando `echo`. Exemplo:

```
$ echo "*"
*
```

```
kali@kali:~/LPIC-1$ echo "*"
*
kali@kali:~/LPIC-1$
```

Exemplo, criação a variável `TESTE=Curso` e vamos usar as aspas duplas para proteger:

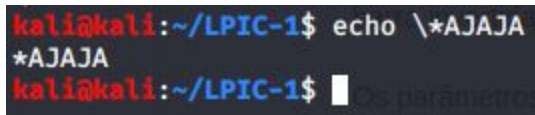
```
kali@kali:~/LPIC-1$ TESTE=Curso
kali@kali:~/LPIC-1$ echo $TESTE
Curso
kali@kali:~/LPIC-1$ echo "$TESTE"
Curso
kali@kali:~/LPIC-1$
```

Note que o CÍFRÃO NÃO FOI IGNORADO e temos o retorno do valor da variável `TESTE`.

Barra Invertida - Escape Character

Uma barra invertida não citada '\ ' é o caractere de escape do Bash. Ele preserva o valor literal do próximo caractere a seguir, com exceção da nova linha. Se um par \newline aparecer e a barra invertida em si não estiver entre aspas, a \newline será tratada como uma continuação de linha (ou seja, será removida do fluxo de entrada e efetivamente ignorada).

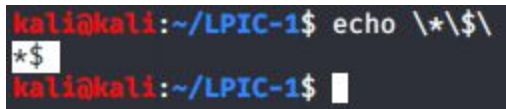
```
$ echo \*AJAJA
```



```
kali@kali:~/LPIC-1$ echo \*AJAJA
*AJAJA
kali@kali:~/LPIC-1$
```

Com vários caracteres, asterisco, cifrão e espaço, por exemplo, para ele proteger cada um desses caracteres:

```
$ echo \*\$\
```



```
kali@kali:~/LPIC-1$ echo \*\$\ 
*$ 
kali@kali:~/LPIC-1$
```