Expressões Regulares, AWK, SED e Wildcards.

Expressões Regulares

Expressão Regular (ou **regex**, abreviação do inglês **regular expression**) provê uma forma concisa e flexível de identificar cadeias de caracteres de interesse, como caracteres particulares, palavras ou padrões de caracteres. Expressões regulares são escritas numa linguagem formal que pode ser interpretada por um processador de expressão regular, um programa que serve um gerador de analisador sintático que examina o texto e identifica as partes que casam com a especificação dada.

Vamos utilizar como ambiente de testes o arquivo **/etc/passwd** , por questões de segurança, **NÃO USE O ROOT**! Se você não sabe se você o **root** ou não, rode o comando abaixo, se for **root** o resultado vai ser **0** , para qualquer resultado diferente de zero, então **você não é root**. Rode no terminal: **id -u**

Agora dê um **cat** no arquivo **/etc/passwd**: **cat /etc/passwd** . A estrutura do arquivo é

login : senha : UID : GID : Nome Completo : Diretório $HOME : shell

Todas as senhas estão com **X**

porque estão com *engenharia reversa* guardadas em outro lugar.

GREP

Em Shell Script geralmente usamos o comando grep para usar expressões regulares. O grep pode ser usado por vários nomes, e cada um deles tem sua particularidade ou nível de evolução: grep, egrep, fgrep e rgrep. Sua sintaxe é definida por: grep [OPÇAO] PADRÃO [ARQUIVO(s)]. Se tratando de expressões regulares, quase tudo na verdade, a melhor forma de aprender é praticando, então, *vamos lá* !

* pegar as linhas que contém determinado PALAVRA (root nesse caso): grep root /etc/passwd
* pegar SOMENTE as linhas que COMEÇAM com determinado PALAVRA (usa-se o ^ antes da PALAVRA): grep '^root' /etc/passwd
* pegar SOMENTE as linhas que TERMINAL com determinado palavra (usa-se o $ DEPOIS da PALAVRA=bash): grep 'bash$' /etc/passwd
* pegar SOMENTE as linhas que CONTÉM PALAVRA que tem duas letras o juntas (ex.: oo) : grep 'oo' /etc/passwd
* pegar SOMENTE as linhas que COMEÇAM com as letras a,e,i,o ou u ([]): grep '^[aeiou]' /etc/passwd
* pegar SOMENTE as linhas que NÃO COMEÇAM com vogais (^[^): grep '^[^aeiou]' /etc/passwd
* pegar SOMENTE as linhas que A SEGUNDA LETRA é uma das letras a,e,i,o ou u(.): grep '^.[aeiou]' /etc/passwd
* pegar SOMENTE as linhas que COMEÇAM com uma VOGAL e TERMINAM com a PALAVRA bash: grep '^[aeiou].*bash$' /etc/passwd
* pegar SOMENTE as linhas que TENHAM EXATOS 32 caracteres (. e {}): egrep '^.{31}$' /etc/passwd

Em alguns comandos usamos egrep e não o grep, porque as chaves fazem parte de um conjunto avançado de Expressões Regulares ("*extended*"), então o egrep lida melhor com elas. Se fosse para usar o grep normal, teria que "*escapar*" as chaves.

Para lista completa de possibilidade use o manual: man grep

AWK

AWK tem a mesma pronúncia de *Auk*, uma *espécie de pássaro*, mas é uma linguagem de programação interpretada que é, geralmente, usada para deixar os scripts de shell com mais recursos. Existem vários tipos de AWK, que nada mais são variantes de outros autores que a modificaram: BWK, GAWK, MAWK, LIBAWK, AWKA, TAWK, JAWK, XGAWK, QSEAWK e BusyBox.Também é possível usar Expressões Regulares com AWK.

Comando básico: echo | awk '{print "Hello, World"}'

Veja que usamos o comando echo junto com awk, para usar o awk devemos nos referir a um arquivo ou usá-la com os comandos setados no arquivo .awk, sendo que o cabeçalho será: /usr/bin/awk . Então vamos criar um arquivo assim:

```
echo -e "Debian\t\tEstável\t\tRápido\t\t10\nSlackware\tPsicopata\tAteu\t\t8\nUbuntu\t\tFácil\t\tSimples\t\t6" > distros.txt
```

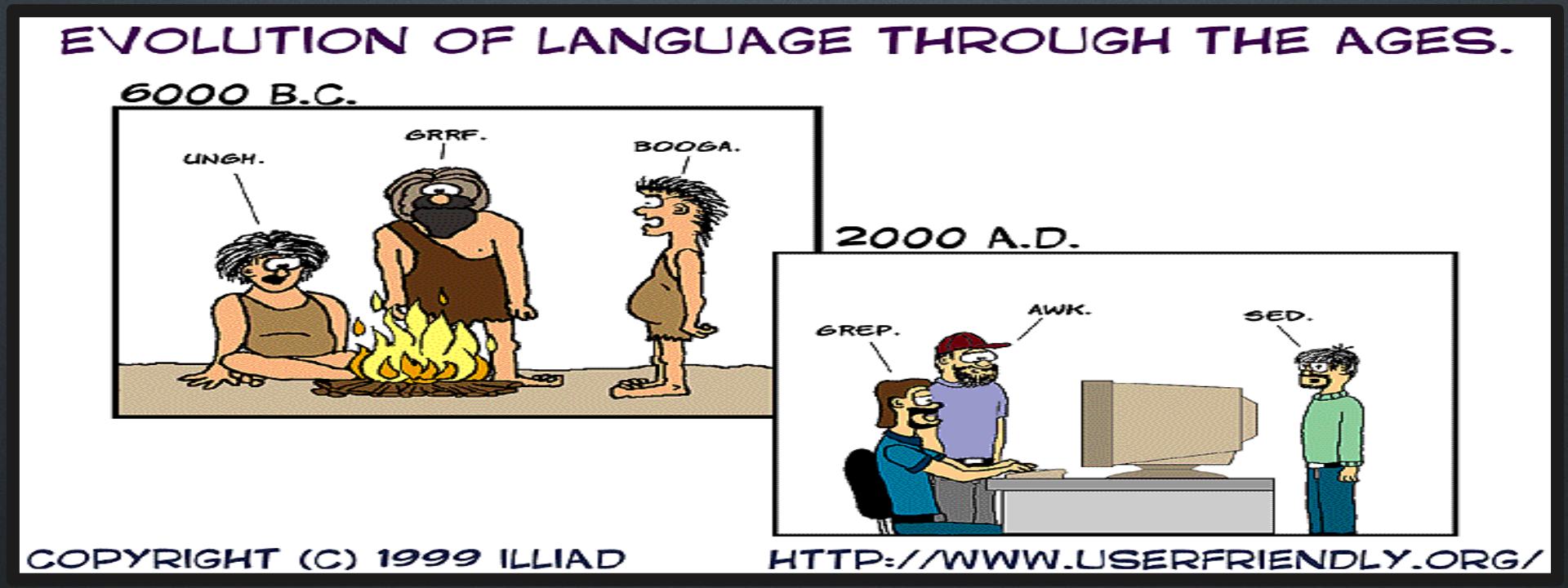
Mostrar só a 1° coluna: awk '{print $1}' distros.txt # 2,3, e 4 ...(...)
Mostrar só a 1° e a 3° colunas: awk '{print $1 $3}' distros.txt # NÃO haverá espaços entre as colunas
Mostrar só a 1° e a 3° colunas com espaços: awk '{print $1, $3}' distros.txt # HAVERÁ espaços entre as colunas
Adicionar informação entre as colunas: awk '{print $1,"é",$2}' distros.txt
Imprimindo texto com quebra de linha \n: awk 'BEGIN {print "Vai ser tudo \n imprimido \n agora\n"}'

Ainda dá pra usar CONDIÇÕES: awk '{ if($4 > 2000) print $1 }' arquivo.txt
REDIRECIONAR SAÍDA: awk 'BEGIN {print 1+1 > "resultado.txt" }'

man awk

SED

sed é o mesmo que **S**tream **ED**itor ou editor de fluxo. Muito utilizado nos sistemas tipo **Unix**. Ao contrário dos editores convencionais, o **sed** atua em linha de comandos ou em **shell script**. Aceita **expressões regulares**, o que lhe confere maior poder, convertendo-se em uma excelente ferramenta para administradores de sistemas.



Sua sintaxe é: **sed** [opções] {script} [arquivo]

Alguns Exemplos de Comandos SED

Vamos usar esse texto de exemplo o arquivo.txt

> O pinguim vive no frio ,mas no meu computador ele se chama tux e seu chão é resfriado pelo cooler e pasta térmica. O tux é o pinguim!

* Troca todas ocorrências da palavra tux pela palavra pinguim: sed 's/tux/pinguim/' arquivo.txt
* Troca ocorrências da palavra pinguim pela palavra tux: sed 's/pinguim/tux/' arquivo.txt
* Imprime só as linhas com a palavra cooler: sed -n '/cooler/p' arquivo.txt
* Deleta só as linhas com a palavra resfriado: sed '/resfriado/d' arquivo.txt
* Retira todas linhas em branco: sed '/^$/d' arquivo.txt

Há uma postagem no blog *terminalroot.com.br* com 30 exemplos do comando sed:

30 exemplos do comando sed (com regex)

http://terminalroot.com.br/2015/07/30-exemplos-do-comando-sed-com-regex.html

Manual do sed : man sed

Glob NÃO é Regex!

Glob	Regex
*.md	.*.md
[aeiou]*	[aeiou].*
filme.{flv,mp4}	filme\.{flv|mp4}
ferias-??.png	ferias-..\.png

O Bash não consegue reconhecer expressões regulares. Dentro de scripts, são comandos e utilitários - como sed e awk - que interpretam REs. O Bash executa a expansão do nome do arquivo - um processo conhecido como glob - mas isso não usa o conjunto padrão de Expressões Regulares.

Em vez disso, globbing reconhece e expande wildcards (em português significa cUringa, cuidado ao pronunciar/escrever , porque é com U e não cOringa com O que é outra coisa). São caracteres que podem ser usados em conjunto com comandos para substituir outros caracteres ou conjunto deles. Eles só pode ser usados com comandos que manipulam arquivos ou diretórios.

Links úteis pra testar REGEX online
http://regexpal.com e http://regex101.com