Linux para Ingeniería:

Shell Scripts

Luis Garreta
luis.garreta@javerianacali.edu.co

Ingeniería de Sistemas y Computación
Pontificia Universidad Javeriana – Cali

28 de marzo de 2017

Programación Shell

- At the heart of Unix is a kernel whose routines aren't easy to use directly
- The 'shell' is the interface between the user and the system.
- Command lines can be put into a file and executed.
- These so-called "shell scripts" can quickly be written and tested.

Tipos de Shells

- Various text-based shells are in use:
  - **sh**, the Bourne Shell, is the oldest.
  - The C-shell (**csh**) has many useful features lacking from sh but isn't that good for programming in.
  - The Korn Shell (**ksh**) and the (very similar) POSIX shell are developments of sh that incorporates many csh features.
  - **bash** is similar and is freely available (it's the default on linux and MacOS X)

Cómo escribir un Script

- Its easy:
  - Start your text editor (vim, nano, emacas, others)
  - Write the instructions (prog1.sh).

```
#!/bin/bash

hostname
echo "La fecha es: " ‘date‘
ls
echo "Adios " $USER
```

  - Save the script with a name with extensión .sh
  - Set executable permissions to the shell
  - Execute:

```
$./prog1.sh
```

Caracterés Comodíng (Wildcard characters) (*)

- * and ? characters have a special meaning to the shell for names.
- * means substitute anything:

```
$ ls *.sh
prog1.sh  prog2.sh

$ ls ar*.gz
archivoTaller.gz  argumentosLinea.gz

$ ls *
prog1.sh prog2.sh archivoTaller.gz  argumentosLinea.gz

$ rm p*

$ rm *.gz
```

Caracterés Comodíng (Wildcard characters) (*)

- * and ? characters have a special meaning to the shell for names.
- **?** means substiture only one character:

```
$ ls prog?.sh
prog1.sh  prog2.sh

$ ls archivoTaller?gz
archivoTaller.gz
```

Argumentos desde la línea de comandos

- Write the script “argumentos.sh”

```
#!/bin/bash

echo Este comando $0 tiene $# argumentos.
echo Estos son $*
```

- Execute the script

```
$ args.sh hola 1 2
echo Este comando args.sh tiene 4 argumentos.
echo Estos son $*
echo El argumento 2 es $1
echo Adios
```

Construcciones

Ciclos e instrucciones de selección

- Ciclos:
    - while, for, do
- Selección:
    - if, case

Ciclo While

```
i=0
while [ $i -lt 10 ]
do
  echo i is $i
  let i=$i+1
done
```

- $i : Valor de la variable i
- -lt : Operador ”Menor que”
- let: asignación de valores a variables

```
i is 0
i is 1
i is 2
i is 3
i is 4
i is 5
i is 6
i is 7
i is 8
i is 9
```

Ciclo While Infinito

```
while true
do
  echo "date is" date
done
```

- Ciclo infinito
- Ctrl+C para terminar

Ciclo For

```
for file in *
do
  echo "wc $file
      gives"
  `wc $file`
done
```

- Itera sobre todos los archivos (*)
- Comillas ‘xxxx‘ ejecuta ese comando

For and If

```
for file in *
do
if [ ! -d $file ]
then
  echo "wc $file
      gives"
  wc $file
else
  echo "$file is a
      directory"
fi
done
```

- Itera sobre todos los archivos (*)
- No cuenta palabras en los que son directorios: if [! -d $file]

Case para múltiples selecciones

```
cd
for file in .?*
do
  case $file in
   .kshrc) echo "You
       have a Korn
       Shell set-up
       file";;
  .bashrc) echo "You
      have a Bash
      Shell set-up
      file";;
  .Xdefaults) echo "
      You have an X
      resource file"
      ;;
  .profile) echo "You
       have a shell
      login file";;
  esac
done
```

- Itera sobre todos los archivos que inician con .
- De acuerdo al nombre imprime el mensaje respectivo

Input/Output Redirección

```
$ date                # A pantalla
$ data > salida.txt        # A un archivo
$ hostname >> out        # Al final de un archivo
$ ls archivoX > /dev/null  # Mensaje de salida a dispositivo
    nulo
$ ls blah 2>/dev/null      # Errores a dispositivo nulo
```

Variables Shell

```
pix=8                         # Valor inicial
let "pix=$pix + 1"            # Asignación de valores
echo $pix                     # Impresion del Valor
```

Arreglos

```
colores[1]=red
colores[2]=green
colores[3]=blue
echo El arreglo colores tiene ${#colores[*]} elementos.
echo Ellos son ${colores[*]}
echo El segundo color es ${colores[2]}
```

Aliases

```
$ alias lr="ls -lr"
$ lr                    # Imprime el listado completo de
    archivos ordenado por fecha
```