



Bash Scripting

Daniel Sheep



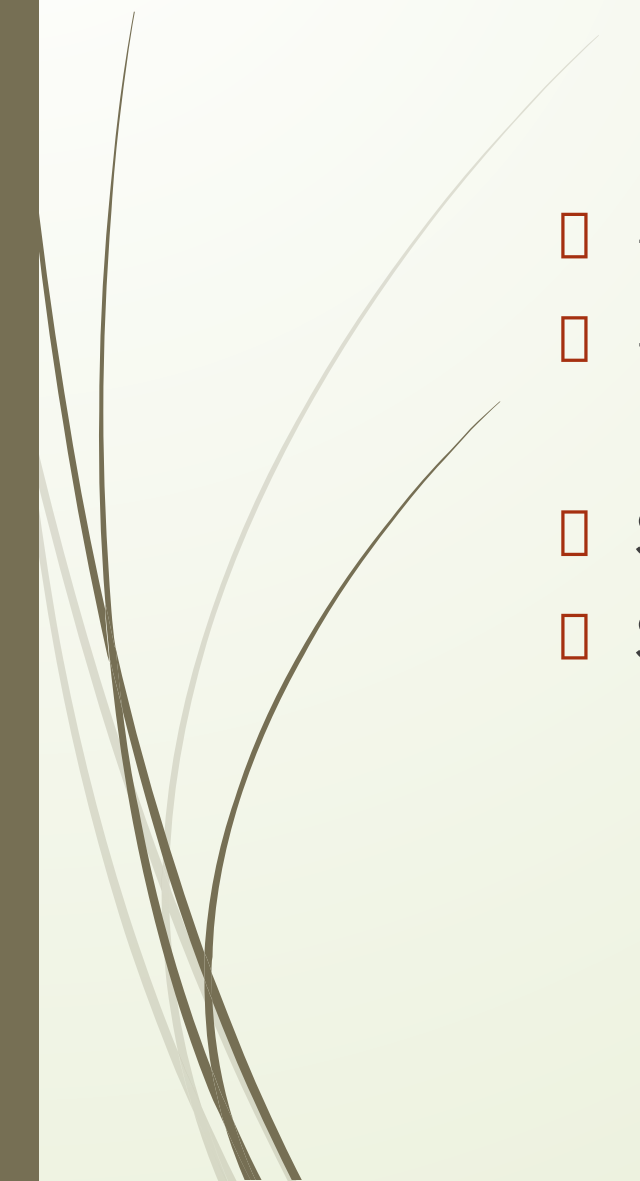
Requerimientos



- ❑ BASH 4.4
- ❑ Masscan
- ❑ Nmap
- ❑ Debian like Linux:
 - ❑ Ubuntu / Kubuntu / Xubuntu
 - ❑ Debian
 - ❑ Kali
- ❑ Fedora like Linux *



Instalar masscan y nmap

- ❑ # apt install masscan
 - ❑ # apt install nmap
 - ❑ \$ sudo install masscan
 - ❑ \$ sudo install nmap
- 

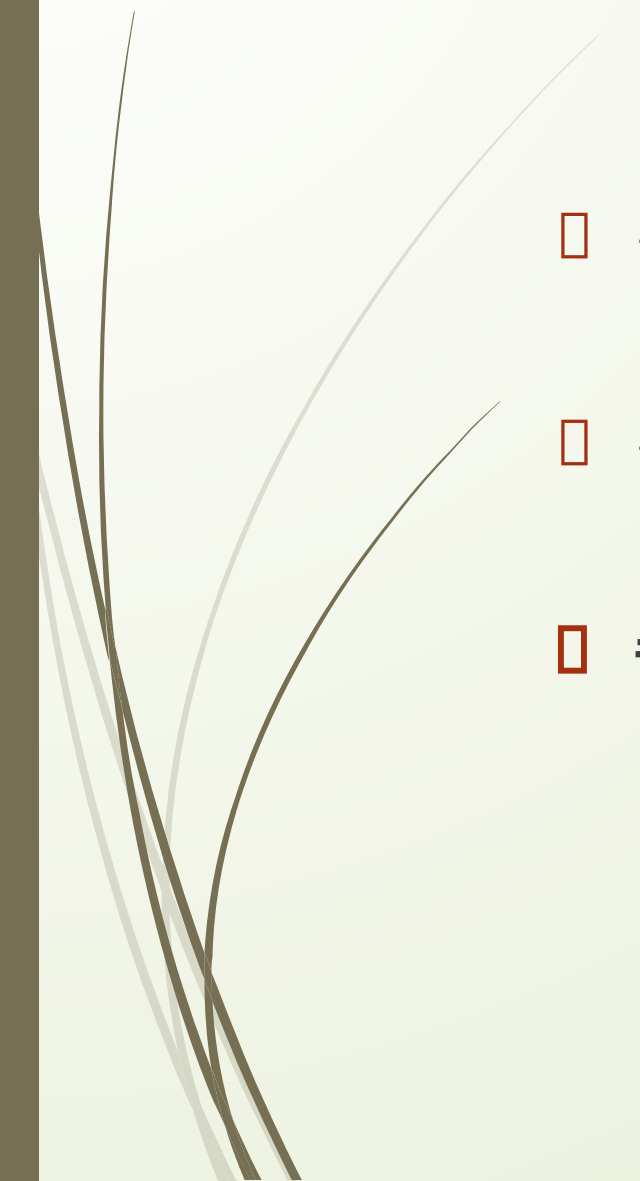


Crear 3 archivos

- ❑ `$ touch script.sh`
- ❑ `$ chmod +x script.sh`
- ❑ `$ touch wtmp messages`
- ❑ `$ head -c 5MB </dev/urandom > wtmp`
- ❑ `$ head -c 6MB </dev/urandom > messages`




Shebang!

- ❑ `#!/usr/bin/awk -f`
 - ❑ `#!/usr/bin/env python`
 - ❑ **`#!/bin/bash`**
- 



Parte 1

- Comentarios
 - Imprimir en pantalla
 - Recursos en dev
 - Redireccionamientos
 - Variables
- 



\$./script.sh

```
#!/bin/bash
```

```
# Variables
```

```
LOG_DIR=/root/Documents/flisol
```

```
#LOG_DIR=/var/log
```

```
clear
```

```
# Clean logs
```

```
cat /dev/null > $LOG_DIR/messages
```

```
cat /dev/null > $LOG_DIR/wtmp
```

```
echo " == Logs cleaned up == "
```

Parte 2

□ Sentencia IF

Operador	Verdad (TRUE) si:

x -lt y	x menor que y
x -le y	x menor o igual que y
x -eq y	x igual que y
x -ge y	x mayor o igual que y
x -gt y	x mayor que y
x -ne y	x no igual que y



Sentencia IF

□ Estructura básica

```
if [[ X condición Y ]]; then  
    TRUE  
fi
```



Parte 3

▣ Funciones

```
# Clean logs
cleaningLOGS(){
    cat /dev/null > $LOG_DIR/messages
    cat /dev/null > $LOG_DIR/wtmp

    echo " == Logs cleaned up == "
}
```



Funciones

- Estructura básica

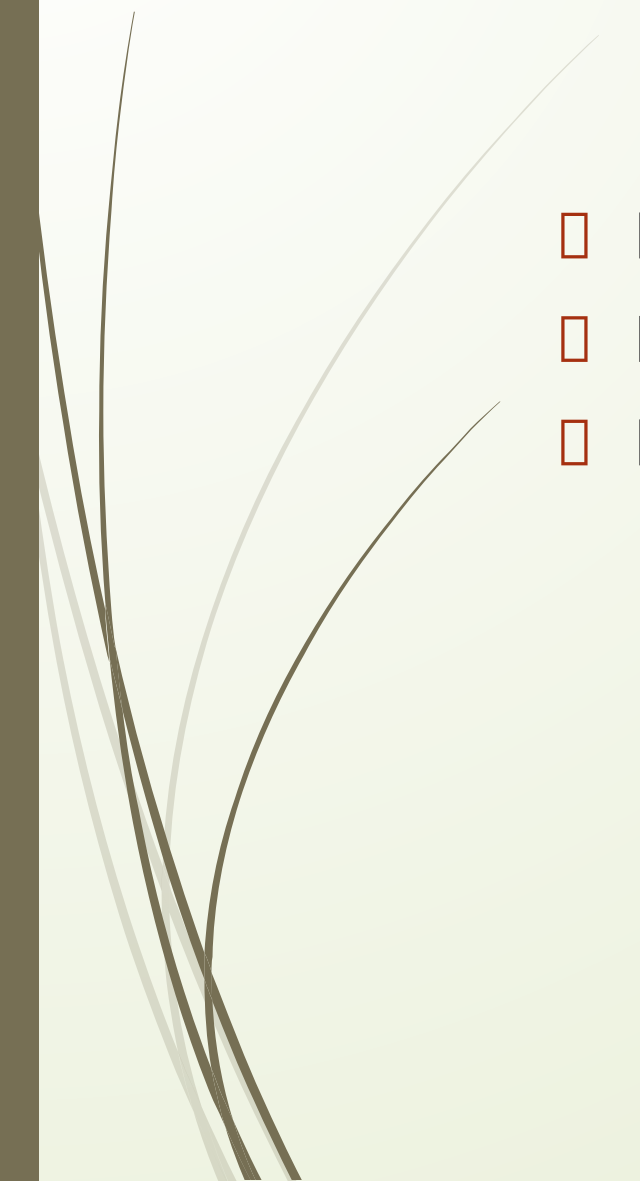
```
func_name () {
```

```
}
```

```
func_name
```



Parte 4

- Entrada de datos
 - Desreferenciado de variables
 - Entubamiento / pipes
- 




Entrada de datos




```
read -p "mensaje: " <variable>
```



Desreferencia de variables



```
$( date +%Y-%m-%d )  
ips_`date +%Y-%m-%d.%H:%M:%S`
```



```
# scanning ports
fastScanNT(){
  read -p " Enter a range of IP {192.168.10.0/24,192.168.2.3} : " target
  rm ips_$(date +%Y-%m-%d).*

  masscan $target --ping --open-only | grep -E -o "([0-9]{1,3}[\.]){3}[0-9]{1,3}" > ips_`date +%Y-%m-%d.%H:%M:%S`

  clear

  echo " == Hosts alive == "
  cat ips_$(date +%Y-%m-%d).*
}
```



Parte 5

- Sentencia while
- Más sobre entubamiento
- Lectura de archivos
- Más sobre variables




Sentencia WHILE

□ Estructura básica

```
While [[ condición ]]; do  
    TRUE  
done
```



Ejemplo de uso



```
cat $IPFILE | ( while read hostname;  
do nmap -sS -Pn --top-ports 10 -oA  
Syn_${hostname} ${hostname}; done;)
```

Cut y Sed

cut: corta cadenas a partir de un delimitador seguido de la fila.

```
cut -d '/' -f1
```

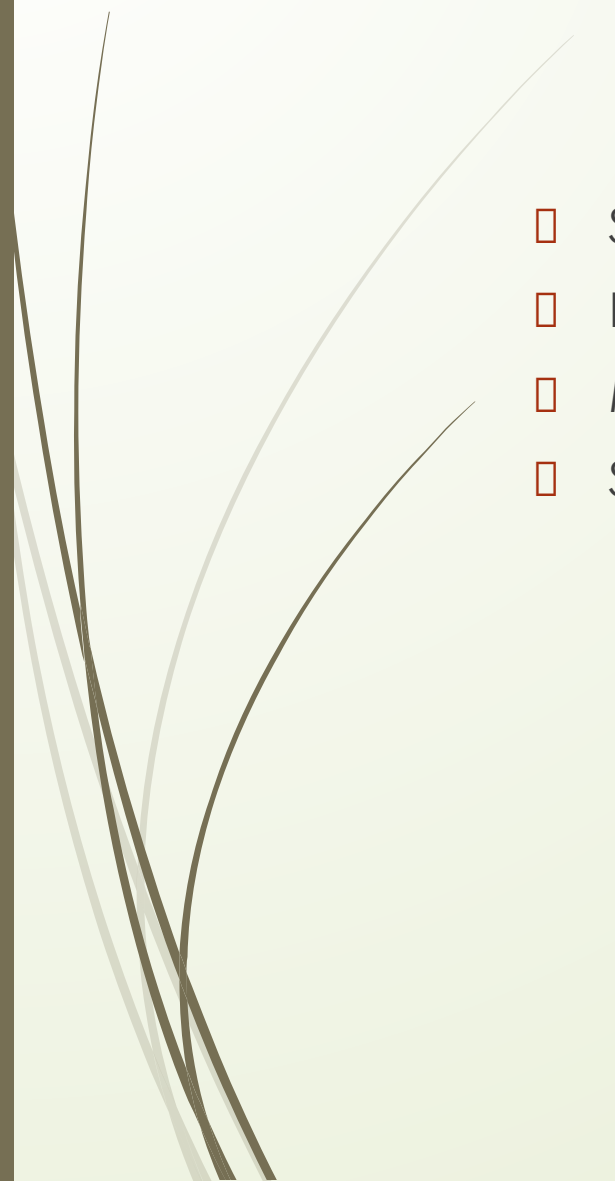
sed: editor de flujo

```
sed ':a;N;$!ba;s/\n/,/g'
```

1. Crea una etiqueta mediante :a
2. Agrega la línea actual y siguiente al espacio del patrón a través de N.
3. Si estamos antes de la última línea, bifurque a la etiqueta creada \$! Ba (\$! Significa no hacerlo en la última línea, ya que debe haber una nueva línea final).
4. Finalmente, la sustitución reemplaza cada línea nueva con un espacio en el espacio del patrón (que es el archivo completo)



Parte 6 y última

- Sentencia switch
 - Repeticiones
 - Menú
 - Sofisticando el script
- 



Menú



```
# Print Menu
printMenu(){
    clear
    echo -e "Introduction to BASH\n"
    echo "  1) Cleaning Logs "
    echo "  2) Fast Network Scan "
    echo -e "    3) Scan Targets \n"
    echo -e "    99) Exit\n"
    echo "IP Address: " $IPADD
}
```




Uso de variables de ambiente

```
isROOT  
if [[ $? -ne 0 ]]; then  
    exit  
fi
```



Sentencia Case / Switch



```
case "$variable" in
  1) sentencias
    ;;
  2) sentencia
    ;;
  ...
  *) default ;;
esac
```