

# Copias de seguridad y control de versiones

¿Cómo tener los cambios en la configuración controlados además de los datos?

## Copias de seguridad

- Copia binaria
  - dd <http://www.thegeekstuff.com/2010/10/dd-command-examples/>
- Copiar archivos y empaquetar
  - cp ; cpio ; tar
- Sincronizar
  - rsync ; rsnapshot
  - <https://help.ubuntu.com/community/BackupYourSystem/TAR>
- Instantáneas
  - LVM snapshots ([http://www.tldp.org/HOWTO/LVM-HOWTO/snapshots\\_backup.html](http://www.tldp.org/HOWTO/LVM-HOWTO/snapshots_backup.html))
- Otros sistemas: p.ej. AMANDA, Bacula, C-Panel, Plesk, etc.
- Reflexión: ¿Copia vs. Backup? <http://www.backupcentral.com/>

# Copias de seguridad

- dd
  - Útil para copiar bit a bit el contenido
  - Ejemplos:
    - dd if=/dev/sda of=/dev/sdb
      - Copia a otro dispositivo
    - dd if=/dev/sda of=~/.hdadisk.img
      - Crea una imagen de sda. Para recuperarla:
        - dd if=hdadisk.img of=/dev/sdb
      - Se pueden especificar particiones: sda1, sda2

# Copias de seguridad

- cpio
  - Copia archivos a y desde (archivos)
    - ls | cpio -ov > /tmp/object.cpio
    - cpio -idv < /tmp/object.cpio
- tar
  - Ejemplos:
    - tar cvzf MyImages-14-09-17.tar.gz /home/MyImages
    - tar -xvf public\_html-14-09-17.tar

<https://help.ubuntu.com/community/BackupYourSystem/TAR>

# Copias de seguridad

- rsync
  - Sincroniza dos una fuente y un destino (incluso a través de SSH)
    - Copia enlaces, dispositivos, propietarios, grupos y permisos
    - Permite excluir archivos
    - Transfiere los bloques modificados de un archivo
    - Puede agrupar todos los cambios de todos los archivos en un único archivo
    - Puede borrar archivos
  - Ejemplos ( <https://rsync.samba.org/examples.html> )
    - `rsync -a --delete source/ destination/`

## Copias de seguridad Ejemplos rsync (I)

- rsync origen destino
  - `rsync /home/Usu1/archiv1[/[/.[.]*] /home/Usu1/archivSecu`
- En un destino remoto
  - Si `echo $RSYNC_RSH=ssh`
    - `rsync /home/Usu1/archiv1/ username@maquina:/rutadestino`
  - Si no
    - `rsync -e "ssh" /home/Usu1/archiv1/ username@maquina:/rutadestino`
- Opciones comunes:
  - `-r` : recursivo ; `-l` : enlaces “simbólicos” ; `-t`: conservar fecha ; `-p`: conservar permisos ; `-o`: conservar propietario ; `-g` : conservar grupo ; `-D` archivos especiales
    - Todas estas opciones incluidas con `-a` , es decir, `-a = -rlptgoD`
  - `-v`: muestra información (vbose)
  - `-z`: comprime
  - `-i`: muestra un resumen final
- **Uso común: `rsync -aviz /home usu@maquina:/backup`**
- ¿Queremos sincronización total? → Borramos archivos: `--delete`
  - `rsync -aviz --delete /home usu@maquina:/backup`
- Restaurando con rsync → invertimos el orden del origen y el destino

# Backups rsnapshot

- script vs. Rsnapshot...
  - [http://www.mikerubel.org/computers/rsync\\_snapshots/](http://www.mikerubel.org/computers/rsync_snapshots/)
  - rsnapshot HOWTO:  
<http://rsnapshot.org/rsnapshot/docs/docbook/rest.html>
- 

## Control de cambios

- Método básico y fundamental
  - Antes de modificar un archivo lo copiamos con otro nombre:
    - `cp /etc/config1 /etc/config.old`
      - old -> (.secu , .vap, etc.)
  - Normalmente en los archivos de configuración podemos escribir comentarios usando #

# Control de cambios

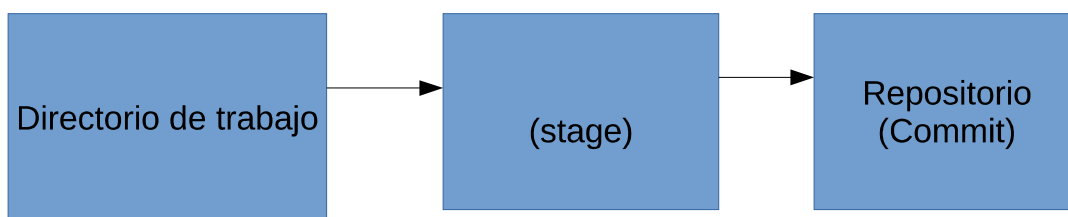
- Hay algunas herramientas que controlan /etc directamente:
  - /etc/keeper
- Usa un sistema de control de versiones por debajo (git)
  - Mejor aprendemos un poco de git y podemos incluir archivos de históricos, logs, etc.
    - Y de scripts de monitorización, configuración, tests, etc.

## Git

- Ventajas (para ISE)
  - Ponemos un pie en devops
  - Seguimiento de los cambios (y de su autor)
  - Permite planificar un entorno de prueba entre varias personas
  - ...
- Ventajas (para su perfil)
  - Demasiadas para enumerar
- Solo veremos los comandos básicos
  - Hay mucha documentación para profundizar

# Cómo funciona

- Directorio de trabajo: lo que tenemos en desarrollo
- Stage: Zona donde se registran los cambios. Ahí incluimos los cambios que queremos seguir.
- Commit: Zona donde los cambios se convierten en permanentes (podemos hacer commits en diferentes ramas “branches”)



## Iniciando un repositorio

- Lo instalamos (si no está)
- Nos situamos en el directorio correspondiente
- Escribimos:
  - `git init`
- Ya hemos tomado la instantánea ¿de qué?
  - `git log`, `git status`, `git branch`
- Cuidado, todo se almacena en el `$PWD/.git`
  - Si borramos el directorio, perdemos el control de cambios...

# Vamos a verlo...

```
alberto@knight rider:~/admscripts$ ls -a
.  ..  PASSWD  SysInfo.py  SysInfo.sh
alberto@knight rider:~/admscripts$ git init
Initialized empty Git repository in /home/alberto/admscripts/.git/
alberto@knight rider:~/admscripts$ ls -a
.  ..  .git  PASSWD  SysInfo.py  SysInfo.sh
alberto@knight rider:~/admscripts$ ls .git
branches  config  description  HEAD  hooks  info  objects  refs
alberto@knight rider:~/admscripts$
```

# Vamos a verlo...

```
alberto@knight rider:~/admscripts$ git log
fatal: your current branch 'master' does not have any commits yet
alberto@knight rider:~/admscripts$
alberto@knight rider:~/admscripts$ git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    PASSWD
    SysInfo.py
    SysInfo.sh

nothing added to commit but untracked files present (use "git add" to track)
alberto@knight rider:~/admscripts$
alberto@knight rider:~/admscripts$ git branch
alberto@knight rider:~/admscripts$
```

# Añadiendo archivos que seguir

- `git add nombre_archivo1 nombre_archivoN`
- Se pueden usar wildcards
  - Cuidado con añadir archivos “delicados”
  - Para asegurarnos, explicitamos en
    - `.gitignore`
- Vamos a ver un ejemplo...

## git add y .gitignore

```
alberto@knighttrider:~/admscripts$ ls
PASSWD SysInfo.py SysInfo.sh
alberto@knighttrider:~/admscripts$ git add SysInfo.sh
alberto@knighttrider:~/admscripts$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   SysInfo.sh

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    PASSWD
    SysInfo.py
alberto@knighttrider:~/admscripts$
```

```
alberto@knighttrider:~/admscripts$ cat > .gitignore
PASSWD
alberto@knighttrider:~/admscripts$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   SysInfo.sh

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    .gitignore
    SysInfo.py
```



# Llevando el primer commit

- Para tener un primer repositorio, añadiremos todos los archivos (incluido el .gitignore) y hacemos el “commit”
- Con la opción -m añadimos un mensaje que explica la modificación
  - Si no lo incluimos, nos llevará a un editor de texto para que lo escribamos

## git commit

```
alberto@knighttrider:~/admscriptss$ git add .
alberto@knighttrider:~/admscriptss$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   .gitignore
    new file:   SysInfo.py
    new file:   SysInfo.sh

alberto@knighttrider:~/admscriptss$ git commit -m "Primer commit de scripts"
[master (root-commit) bd87bc0] Primer commit de scripts
 3 files changed, 29 insertions(+)
 create mode 100644 .gitignore
 create mode 100755 SysInfo.py
 create mode 100755 SysInfo.sh
alberto@knighttrider:~/admscriptss$ git status
On branch master
nothing to commit, working tree clean
alberto@knighttrider:~/admscriptss$
```

# Modificando un *contenido*

- Modificamos SysInfo.sh, vamos a añadir una línea (en rojo) y lo añadimos al stag (git add SysInfo.sh)

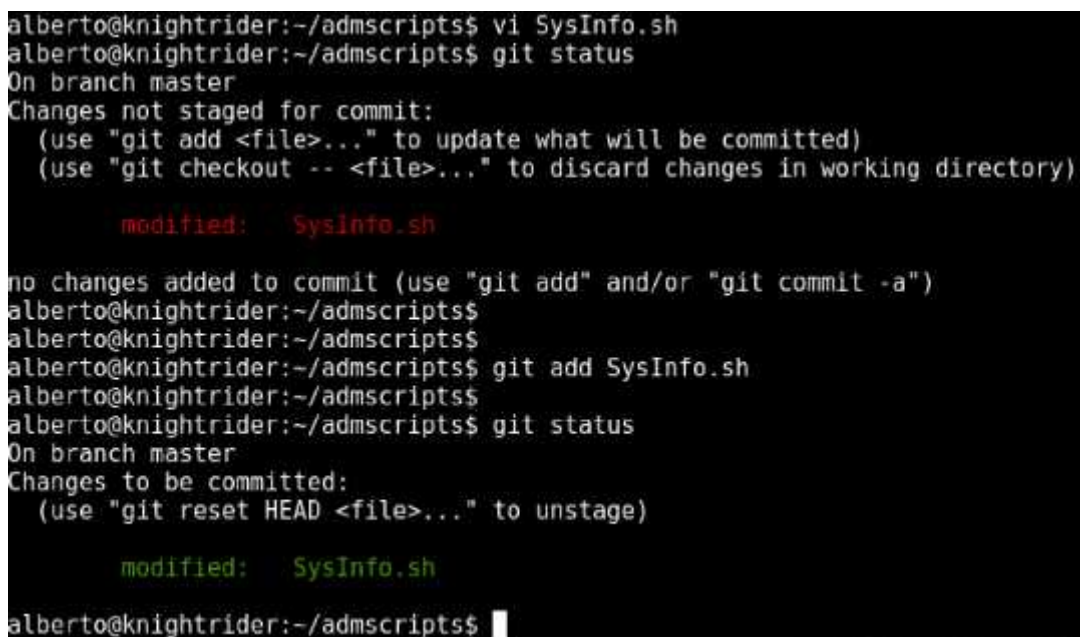
```
#!/usr/bin/env bash
#A System Information Gathering Script

#Command 1
UNAME="uname -a"
printf "Gathering system information with the $UNAME command: \n\n"
$UNAME

#Command 2
DISKSPACE="df -h"
printf "Gathering disk space information with the $DISKSPACE command: \n\n"
$DISKSPACE

printf "Informe generado el día "; echo `date`
```

## Modificando contenido



```
alberto@knighttrider:~/admscripts$ vi SysInfo.sh
alberto@knighttrider:~/admscripts$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   SysInfo.sh

no changes added to commit (use "git add" and/or "git commit -a")
alberto@knighttrider:~/admscripts$
alberto@knighttrider:~/admscripts$
alberto@knighttrider:~/admscripts$ git add SysInfo.sh
alberto@knighttrider:~/admscripts$
alberto@knighttrider:~/admscripts$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   SysInfo.sh

alberto@knighttrider:~/admscripts$
```

# Viendo las modificaciones: diff

- Con diff, podemos ver qué diferencias hay entre lo que se ha modificado en el directorio de trabajo y lo que está siendo seguido en el stag
- Si modif y luego add → diff==0
- Si add y luego modif → diff muestra

# Viendo las modificaciones: diff

```
alberto@knighttrider:~/admscripts$ git reset HEAD SysInfo.sh
Unstaged changes after reset:
M       SysInfo.sh
alberto@knighttrider:~/admscripts$ git diff
diff --git a/SysInfo.sh b/SysInfo.sh
index 397aade..fe9f654 100755
--- a/SysInfo.sh
+++ b/SysInfo.sh
@@ -11,3 +11,4 @@ DISKSPACE="df -h"
 printf "Gathering disk space information with the $DISKSPACE command: \n\n"
 $DISKSPACE

+printf "Informe generado el día "; echo `date`
alberto@knighttrider:~/admscripts$
alberto@knighttrider:~/admscripts$ git add SysInfo.sh
alberto@knighttrider:~/admscripts$
alberto@knighttrider:~/admscripts$ git diff
alberto@knighttrider:~/admscripts$
```

Hemos hecho el reset del HEAD para el archivo puesto que Ya lo habíamos añadido antes...

# Llevando modificación al repositorio

- Una vez que queremos dejar las modificaciones como definitivas (llevarlas al repositorio), usamos git commit
- Si no hemos añadido “a mano” con git add los archivos, git commit -a lo hará por nosotros

# Llevando modificación al repositorio

```
alberto@knighttrider:~/admscripts$ git commit -m " Añadido fecha en bash "
```

[master 3c33539] Añadido fecha en bash  
1 file changed, 1 insertion(+)

```
alberto@knighttrider:~/admscripts$ git status
```

On branch master  
nothing to commit, working tree clean

```
alberto@knighttrider:~/admscripts$ git log
```

commit 3c33539528fe2243e00e190c5b66924a781dabcd  
Author: alberto <aguillen@ugr.es>  
Date: Sat Sep 16 14:35:30 2017 +0200

Añadido fecha en bash

```
commit bd87bc073d0aefa55f9305502b57d9fc5578eae5
```

Author: alberto <aguillen@ugr.es>  
Date: Sat Sep 16 14:20:03 2017 +0200

Primer commit de scripts

```
alberto@knighttrider:~/admscripts$
```

# Recuperando el contenido

- Antes de un commit:
  - Exploramos un varias formas de formato de fecha pero nos quedamos con la original, ¿Cómo podemos recuperar lo que teníamos?
    - `git checkout HEAD nomb_archivo`
      - Devolverá el nombre de archivo a la última versión del commit
    - HEAD es un puntero al último commit hecho, aunque puede retroceder ¿cómo? con `git reset`
  - Tras un commit
    - Podemos recuperar un commit concreto usando:
      - `git reset 7_primeros_caracteres_del_commit_SHA`
        - ¡Actualizará el HEAD!
  - Hay otras formas de trabajar (workflows), p.ej., implementar una funcionalidad nueva usando una rama nueva (branch) y luego uniendo con otra rama o la principal (merge)...



## Recuperando contenido

```
alberto@knighttrider:~/admscripts$ git status
On branch master
nothing to commit, working tree clean
alberto@knighttrider:~/admscripts$ git log
commit 3c33539528fe2243e00e190c5b66924a781dabcd
Author: alberto <aguillen@ugr.es>
Date: Sat Sep 16 14:35:30 2017 +0200
```

Añadido fecha en bash

```
commit bd87bc073d9aefa55f9305502b57d9fc5578eae5
Author: alberto <aguillen@ugr.es>
Date: Sat Sep 16 14:20:03 2017 +0200
```

Primer commit de scripts

```
alberto@knighttrider:~/admscripts$ git show
commit 3c33539528fe2243e00e190c5b66924a781dabcd
Author: alberto <aguillen@ugr.es>
Date: Sat Sep 16 14:35:30 2017 +0200
```

Añadido fecha en bash

```
diff --git a/SysInfo.sh b/SysInfo.sh
index 397aade..fe9f654 100755
--- a/SysInfo.sh
+++ b/SysInfo.sh
@@ -11,3 +11,4 @@ DISKSPACE="df -h"
 printf "Gathering disk space information with the $DISKSPACE"
+printf "Informe generado el día "; echo `date`
alberto@knighttrider:~/admscripts$
```

```
alberto@knighttrider:~/admscripts$ vi SysInfo.sh
alberto@knighttrider:~/admscripts$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)
```

modified: SysInfo.sh

```
no changes added to commit (use "git add" and/or "git commit -a")
alberto@knighttrider:~/admscripts$ git checkout HEAD SysInfo.sh
alberto@knighttrider:~/admscripts$ git status
On branch master
nothing to commit, working tree clean
alberto@knighttrider:~/admscripts$
```



# Recuperando contenido (II)

Modificamos dos veces y hacemos commit

```
alberto@knighttrider:~/admscripts$ git status
On branch master
nothing to commit, working tree clean
alberto@knighttrider:~/admscripts$ sed s/'date'/'date +%A %B %y'/' -i SysInfo.sh
alberto@knighttrider:~/admscripts$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   SysInfo.sh

no changes added to commit (use "git add" and/or "git commit -a")
alberto@knighttrider:~/admscripts$ git commit -a
[master 0765828] modificamos formato fecha
 1 file changed, 1 insertion(+), 1 deletion(-)
alberto@knighttrider:~/admscripts$ sed s/'date'/'date +%A %B %y'/' -i SysInfo.sh
alberto@knighttrider:~/admscripts$ git commit -a -m "Mejoramos presentación"
On branch master
nothing to commit, working tree clean
alberto@knighttrider:~/admscripts$ sed s/'date +%A %B %y'/'date +%A %B %y' -i SysInfo.sh
alberto@knighttrider:~/admscripts$ git commit -a -m "Mejoramos presentación"
[master f1a7721] Mejoramos presentación
 1 file changed, 1 insertion(+), 1 deletion(-)
```

```
alberto@knighttrider:~/admscripts$ git log
commit f1a77210c495365d6a81298575c5d02801289994
Author: alberto <aguillen@ugr.es>
Date: Sun Sep 17 01:06:47 2017 +0200

    Mejoramos presentación

commit 0765828256f850ee2ec06a3e9cdd11a18fdab863
Author: alberto <aguillen@ugr.es>
Date: Sun Sep 17 01:03:09 2017 +0200

    modificamos formato fecha

commit 3c33539528fe2243e00e190c5b66924a781dabcd
Author: alberto <aguillen@ugr.es>
Date: Sat Sep 16 14:35:30 2017 +0200

    Añadido fecha en bash

commit bd87bc073d0aefa55f9305502b57d9fc5578eae5
Author: alberto <aguillen@ugr.es>
Date: Sat Sep 16 14:20:03 2017 +0200

    Primer commit de scripts
```

```
alberto@knighttrider:~/admscripts$ git show 0765828256f850ee2ec06a3e9cdd11a18fdab863
commit 0765828256f850ee2ec06a3e9cdd11a18fdab863
Author: alberto <aguillen@ugr.es>
Date: Sun Sep 17 01:03:09 2017 +0200

    modificamos formato fecha

diff --git a/SysInfo.sh b/SysInfo.sh
index fe9f654..2797c16 100755
--- a/SysInfo.sh
+++ b/SysInfo.sh
@@ -11,4 +11,4 @@ DISKSPACE="df -h"
 printf "Gathering disk space information with the $DISKSPACE command: \n\n"
 $DISKSPACE

-printf "Informe generado el día "; echo `date`
+printf "Informe generado el día "; echo `date +%A %B %y`
```

Primer commit de scripts

Listado de commits

Detalle de un commit

# Recuperando contenido (III)

```
alberto@knighttrider:~/admscripts$ git reset 3c33539
Unstaged changes after reset:
M   SysInfo.sh
alberto@knighttrider:~/admscripts$ git diff
diff --git a/SysInfo.sh b/SysInfo.sh
index fe9f654..52f384a 100755
--- a/SysInfo.sh
+++ b/SysInfo.sh
@@ -11,4 +11,4 @@ DISKSPACE="df -h"
 printf "Gathering disk space information with the $DISKSPACE command: \n\n"
 $DISKSPACE

-printf "Informe generado el día "; echo `date`
+printf "Informe generado el día "; echo `date +%A %B %y`
```

Recuperamos el repositorio  
pero el directorio de trabajo no  
→ Hay que hacer checkout

Al hacer reset, perdemos commits

```
alberto@knighttrider:~/admscripts$ git reset --hard 3c33539
HEAD is now at 3c33539 Añadido fecha en bash
alberto@knighttrider:~/admscripts$ git diff
alberto@knighttrider:~/admscripts$ git status
On branch master
nothing to commit, working tree clean
alberto@knighttrider:~/admscripts$ cat SysInfo.sh | grep date
printf "Informe generado el día "; echo `date`
alberto@knighttrider:~/admscripts$
```

```
alberto@knighttrider:~/admscripts$ git log
commit 3c33539528fe2243e00e190c5b66924a781dabcd
Author: alberto <aguillen@ugr.es>
Date: Sat Sep 16 14:35:30 2017 +0200

    Añadido fecha en bash

commit bd87bc073d0aefa55f9305502b57d9fc5578eae5
Author: alberto <aguillen@ugr.es>
Date: Sat Sep 16 14:20:03 2017 +0200

    Primer commit de scripts
```

Con la opción --hard, restaura repositorio y directorio de trabajo

# Trabajo con varias máquina (o personas): remote

- Añadimos origen remoto de los datos
  - (especificar protocolo http, git, sftp)
  - git ADD origin
  - git remote -v
- 

## Enlaces para saber más

- Documentación oficial
  - <https://git-scm.com/docs/gittutorial>
- Curso codecademy (es práctico)
- Github:
  - <https://try.github.io/levels/1/challenges/1>
- Integración con bash:
  - <https://git-scm.com/book/es/v2/Git-in-Other-Environments-Git-in-Bash>