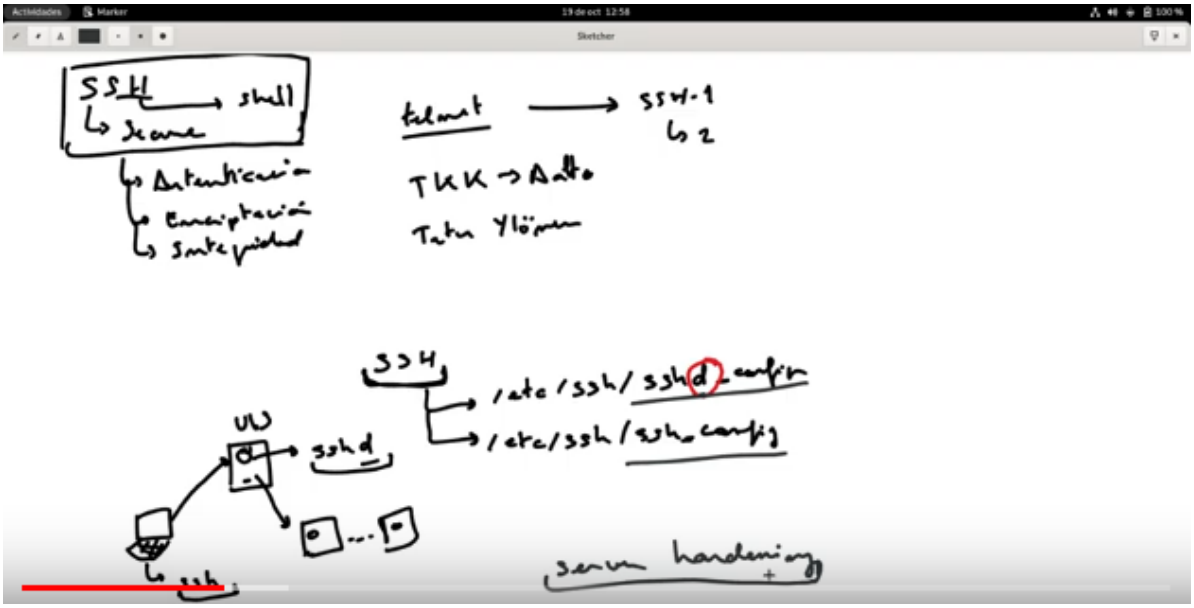
ISE 22/23: Práctica 2

Vídeo 1 - P2L1 Configuración de SSH Pt I



Ubuntu server:

```
NO PERMITIR ROOT ------------------------------------------------------------------
-

>ip add     //Ver nuestra IP

>apt search ssh
>apt search ssh | grep server
>sudo apt install openssh-server    //ERROR: no encuentra IP
>sudo apt update
>sudo apt install openssh-server
>ps -Af | grep sshd     //Para ver si el servicio se ha iniciado directamente -
SI

>ssh localhost
    Primera vez nos muestra la fingerprint del servidor - SI
    Autenticamos...
    CTRL+D salimos
>ls -la     //Ahora hay un directorio ssh, dentro está el fingerprint de
localhost
>cd .ssh/

Vamos a deshabilitar el acceso de root porque eso es un problema de seguridad
>sudo vi /etc/ssh/sshd_config
```

> ```
> OLD: PermitRootLogin prohibit-password //Deshabilitada por defecto
> NEW: PermitRootLogin no
> ```

```
>sudo systemctl restart sshd
    En Ubuntu es suficiente con ssh, pero en rocky no, así que por consistencia vamos siempre a hacer la distinción ssh/sshd

Comprobamos si funciona intentando acceder con root desde otra terminal externa:
>>ssh -l root 192.168.56.15 //IP del ubuntu server -- ERROR pemission denied
>>ssh root@192.168.56.15 -v //Otra forma de hacerlo, con verbose activado


CAMBIAR EL PUERTO ------------------------------------------------------------
>sudo vi /etc/ssh/sshd_config -- Descomentar y cambiar puerto a 22022
>systemctl restart sshd

>>ssh 192.168.56.15 -l alberto -p 22022 //Nos deja sin problema, porque el firewall en ubuntu está desactivado por defecto - vamos a activarlo y añadir el puerto

ACTIVAR FIREWALL -------------------------------------------------------------
>sudo ufw status
>sudo ufw enable

>>ssh 192.168.56.15 -l alberto -p 22022 //ERROR

>sudo ufw allow 22022
>>ssh 192.168.56.15 -l alberto -p 22022
```

CentOS

```
NO PERMITIR ROOT -------------------------------------------------------------
>ps -Af | grep sshd
    SSH si que está en la versión por defecto de CentOS, no hay que instalar como en Ubuntu (aunque durante la instalación nos pregunta si lo queremos)
>systemctl status sshd
>ssh localhost                  //CTRL+D salir

Prueba desde fuera:
>>ssh 192.168.56.10 -l alberto  //Acceso correcto. Y root?
>>ssh 192.168.56.10 -l root     //Se puede SIN CONTRASEÑA. Hay que cambiarlo

>sudo vi /etc/ssh/sshd_config
```

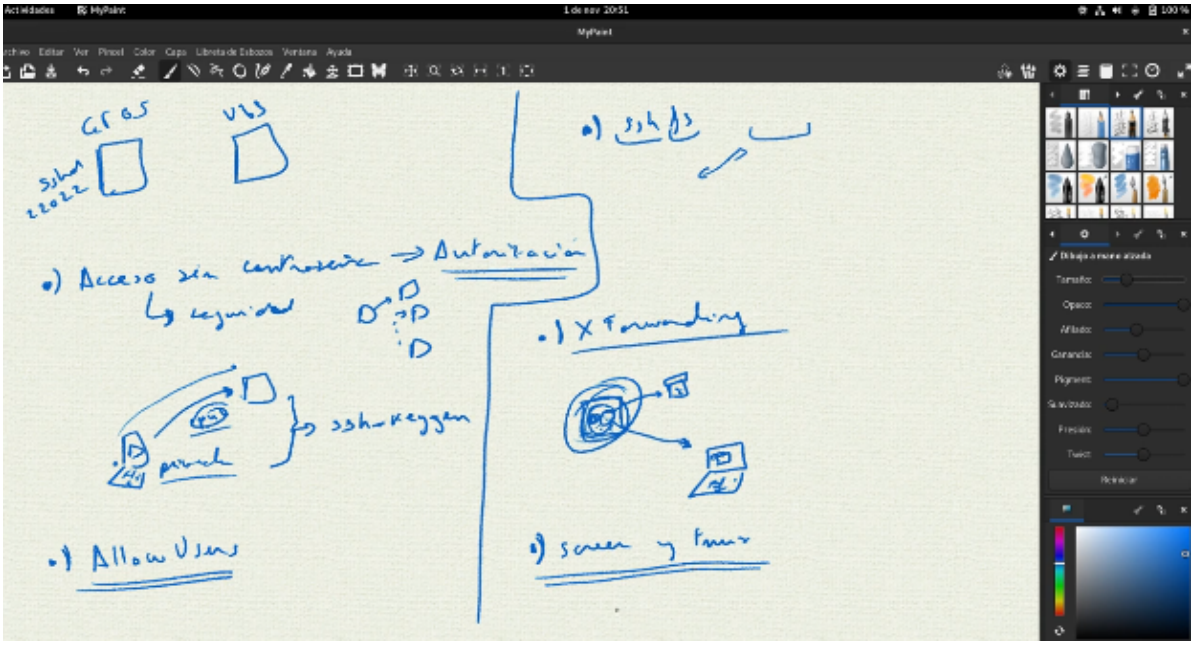
> ```
> OLD: PermitRootLogin yes
> NEW: PermitRootLogin no
> ```

```
Comprobar:
>>ssh 192.168.56.10 -l root     //Nos ha dejado! No hemos reiniciado el servicio
>systemctl restart sshd
>>ssh 192.168.56.10 -l root     //Ya no nos deja
```

```
CAMBIAR EL PUERTO ------------------------------------------------------------
-
>sed s/'Puerto 22'/'Puerto 22022'/ -i /etc/ssh/sshd_config
>systemctl restart sshd

>>ssh 192.168.56.10 -l alberto -p 22022 //No deja? Mira que no esté comentada la
línea. Descomentamos en el config
>systemctl restart sshd             //ERROR
>systemctl status sshd
>journalctl -xe
    Resulta que no tenemos permiso, si leemos bien el config resulta que hay una
línea avisándonos sobre esto y lo que tenemos que hacer

>dnf provides semanage  //ver qué paquete proporciona el comando
>dnf install [nombre del paquete, policycoreutils....]
>semanage port -l | grep ssh        //ver tipos de puertos relacionados con ssh
>semanage por -a -t ssh_port_t -p tcp 22022
>semanage port -l | grep ssh

>systemctl restart sshd
>>ssh 192.168.56.10 -l alberto -p 22022     //ERROR
>ssh localhost -p 22022         //Si va, así que está proporcionando servicio
    El problema es que en la terminal externa nos está bloqueando el firewall

>sudo firewall-cmd --add-port 22022/tcp --permanent //Opcional permanente -- ojo
que si se hace permanente, luego tenemos que hacer
    >sudo firewall-cmd --add-port 22022/tcp
    ó
    >sudo firewall-cmd --reload

>>ssh 192.168.56.10 -l alberto -p 22022     //We're in
```

	Instalación SSH	**Nomenclatura**	**PermitRootLogin**	**Firewall**	**Firewall activado por defecto**	
Ubuntu server	hay que instalarlo / pregunta durante la instalación	ssh y sshd	prohibit-password	ufw	No	
CentOS	por defecto	solo sshd	yes	firewall-cmd	Si	

Vídeo 2 - P2L3 Configuración de SSH Pt II sshfs, -X, screen



CentOS

```
CREAR CLAVE ------------------------------------------------------------------
El tema de la clave pública y privada se hace para autorización. Para cifrado
usamos clave simétrica
>ssh-keygen
    Nos pregunta por el directorio y si queremos contraseña o no, en este caso
no.
>ls -la .ssh/
    Vemos que tenemos clave privada (lectura y escritura solo para nosotros) y
clave pública (RW para otros grupos)

COPIAR CLAVE -----------------------------------------------------------------
>ssh-copy-id 192.168.56.15 -p 22022
>ssh 192.168.56.15 -p 22022
    Hemos logrado iniciar sesión en la máquina de ubuntu server sin contraseña
```

Ubuntu Server

```
>sudo vi /etc/ssh/sshd_config
```

> ```
> OLD: #PasswordAuthentication yes
> NEW: PasswordAuthentication no
> ```

```
>sudo systemctl restart sshd
```

CentOS

```
>ssh 192.168.56.15 -p 22022
    Hemos logrado iniciar sesión. Y desde una consola externa?
>>ssh 192.168.56.15 -p 22022
    No podemos: no quedan alternativas para acceder. Tendríamos que volver a
cambiar el config en Ubuntu y permitir contraseñas, copiar la clave de la
terminal externa con ssh-copy-id y luego volver a quitar el acceso con contraseña
y reiniciar el servicio.

ALLOW USERS ------------------------------------------------------------------
>sudo adduser nico
>sudo passwd nico
>ssh 192.168.56.15 -l nico -p 22022 //acordarse de PasswordAuthentication en
config en US -- También crear user en US:
    >sudo adduser nico
>ssh 192.168.56.15 -l nico -p 22022
```

Ubuntu Server

```
Solo vamos a permitir a alberto:
>sudo vi /etc/ssh/sshd_config
```

> Añadir línea:
>
> ```
> AllowUsers alberto
> ```

```
>systemctl restart ssh
```

CentOS

```
>ssh 192.168.56.15 -l nico -p 22022 //ERROR
>ssh 192.168.56.15 -l alberto -p 22022

SSHFS ------------------------------------------------------------------
Filesystem, acceso a información remota garantizando la seguridad
>dnf provides sshfs
>sudo dnf install [nombre del paquete fuse-....]

>mkdir ./ubuntuserver
>sshfs 192.168.56.15:/home/alberto ./ubuntuserver/ -p 22022
>mount
>cd ubuntuserver ls -la //Hacer touch a lo que sea desde US y volver a comprobar
que sí aparece en el directorio local

X FORWARD --------------------------------------------------------------------
Ejecución en el servidor desde la interfaz del cliente

>ssh 192.168.56.15 -p 22022 -X
>sudo apt get gedit
>gedit hola.txt //Vemos la interfaz normal pero la ejecución se hace en US, y se
guardarán las cosas en US - Comprobar con ls
P.ej podríamos ejecutar el navegador de esta forma para acceder a contenidos
restringidos por ip, como si fuera una vpn
```

```
SCREEN Y TMUX ------------------------------------------------------------------
Lanzar un trabajo en una terminal y dejarlo ejecutando sin sesión iniciada para
poder retomarlo luego

>ssh 192.168.56.15 -p 22022
>vi miarchivo
```

Ubuntu Server

```
>ps afx
    Vemos los procesos: ssh, bash, vi en la otra terminal... Si cerramos la
terminal de centOS repentinamente, se cierra la sesión y se cierran en cascada
todos los procesos de vi, perdiéndose lo que hayamos tocado

>ps afx | grep vi
```

CentOS

```
>screen
>vi miotroarchivo
```

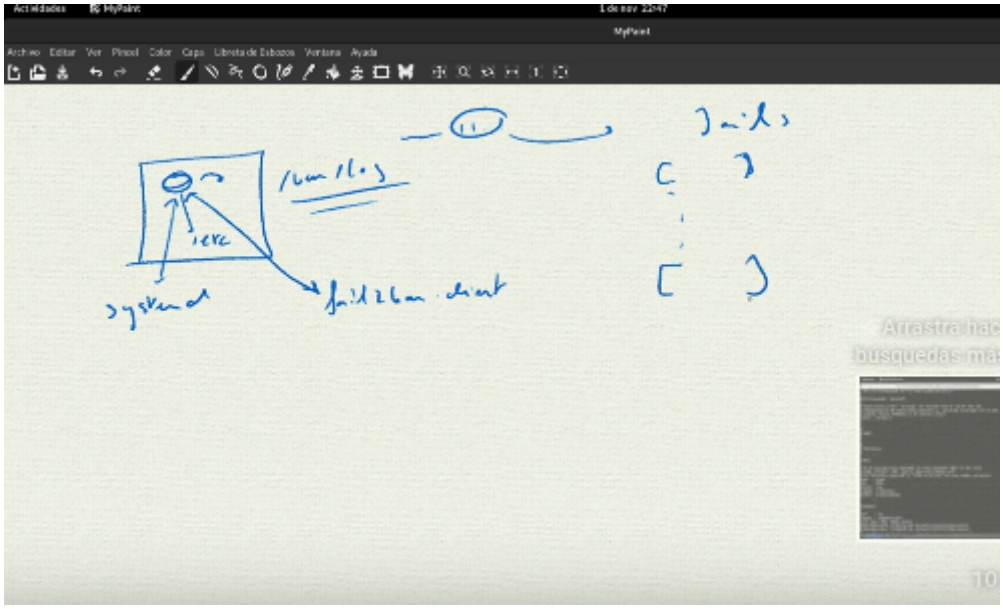
Ubuntu Server

```
>ps afx
    Vemos como de screen cuelga bash, vi... Si cerramos repentinamente en centOS,
seguirá screen con sus procesos hijo.
```

CentOS

```
Para recuperar la sesión de screen:
>screen -r -d
    Este comando también lo podemos ejecutar desde el servidor en US y lo
recuperaría igual de bien.
    CTRL+A CTRL+D Detach manual

>screen -list

TMUX misma funcionalidad que screen. CTRL+D pantallas verticales. Comprobar en
server con ps afx los procesos. CTRL+B CTRL+D Detach manual
```

Vídeo 3 - P2L3 Configuración de SSH Pt III Fail2ban



CentOS

```
FAIL2BAN ------------------------------------------------------------------
Bloquear direcciones IP tras intentos fallidos de login - evitamos ataques de
fuerza bruta

>dnf search fail2ban    //No hay coincidencias
>dnf search epel        //Cjto de paquetes extendido
>sudo dnf install epel-release
>dnf search fail2ban
>sudo dnf install fail2ban

>systemctl status fail2ban  //Deshabilitado e inactivo
>systemctl enable fail2ban  //Habilitar para que en siguiente reinicio se active
>systemctl start fail2ban   //Iniciar manualmente

Ahora lo vamos a configurar:
>cd /etc/fail2ban
>less jail.conf
    Nos dice en mayus cómo activar las cárceles y que no modifiquemos el archivo,
porque es el archivo por defecto y si viene una actualización se sobreescribiría
lo que guardáramos ahí. Así que haremos una copia local.
>sudo cp -a jail.conf jail.local
>sudo vi jail.local
```

> ```
> [sshd]
> ...
> enabled = true
> ```

```
>sudo fail2ban-client status sshd   //No encuentra la jail
>sudo systemctl restart fail2ban.service
>sudo fail2ban-client status sshd   //Está operativa

>> ssh 192.168.56.10 -p 22022       //Introducir pass errónea varias veces (5)
>sudo fail2ban-client status sshd
    En el estado de la cárcel vemos los intentos. 5 por defecto. Vemos que ha baneado la ip.
>> ssh 192.168.56.10 -p 22022       //Introducir pass correcta
    Nos loguea aunque pusiera que estábamos baneados, ¿por qué? No habíamos modificado el puerto.
>sudo vi jail.local
```

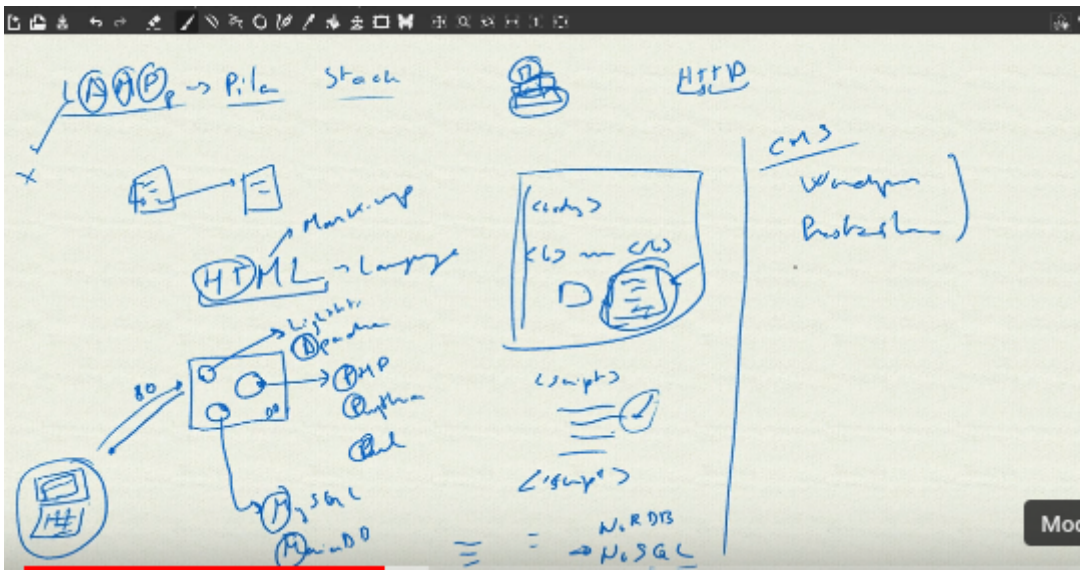
```
[sshd]
...
port = 22022
```

```
>sudo systemctl restart fail2ban.service
>> ssh 192.168.56.10 -p 22022       //No nos deja, estamos baneados


Para sacar una IP de la lista de baneados:
>sudo fail2ban-client set sshd unbanip 192.168.56.1
>sudo fail2ban-client status sshd
    El contador no se actualiza por un bug, pero la ip ya no aparece en la lista de baneados.
>> ssh 192.168.56.10 -p 22022       //Entramos correctamente
```

Vídeo 4 - P2L3 LAMP

HTML -> HyperText Markup Language

LAMP -> Linux Apache MySQL PHP



CentOS

```
>dnf search apache
>sudo dnf install httpd
>systemctl status httpd         //Deshabilitado e inactivo
>sudo systemctl enable httpd
>sudo systemctl start httpd

>curl localhost

>dnf search php
>sudo dnf install php
>php -a         //php es un intérprete, no un servicio. No se mira en systemctl

>dnf search mariadb
>sudo dnf install mariadb
>systemctl status mariadb       //ERROR: solo hemos instalado el cliente
>sudo dnf install mariadb-server
>systemctl status mariadb
>sudo systemctl enable mariadb
>sudo systemctl start mariadb

>mysql -u root
    ojito que nos deja entrar sin pass - hay que ejecutar el script que nos
recomienda la documentación y el manual:
>mysql-secure-installation      //Opciones por defecto
>mysql -u root                  //Ahora da error porque falta la pass
>mysql -u root -p

>>curl 192.168.56.10
>>ping 192.168.56.10
    Curl nos da error, pero con Ping vemos que si hay conexión - el problema es
el firewall
>sudo firewall-cmd --add-port=80/tcp
>sudo firewall-cmd --add-port=80/tcp --permanent
>sudo firewall-cmd --reload
>>curl 192.168.56.10
```

Buscamos en internet el script de ejemplo de mysqli_connect - Lo copiamos en nuestro equipo centOS:

```
>les /etc/httpd/conf/httpd.conf
    Vemos que los archivos se guardan en /var/www/html. Ahí irá el script
>sudo vi index.php              //Copiar aquí el script
```

> [Sobra decir que esto son malas prácticas, en realidad no deberíamos nunca hardcodear esta info en nuestro index.php, sino tenerla en algún tipo de archivo protegido]
>
> La ip se puede quedar como está
>
> En el usuario ponemos root
>
> En la contraseña practicas,ISE

> La bd vamos a crearla ahora mismo:

```
>mysql -u root -p
    CREATE DATABASE mi_bd
```

Desde un navegador vamos a 192.168.56.10 - vemos una página por defecto

Si vamos a 192.168.56.10/index.php nos comemos una mierda, solo se ve el texto. Pasa que php no está interpretando el archivo.

```
>sudo vi /etc/httpd/conf/httpd.conf
```

> ```
> OLD: DirectoryIndex index.html
> NEW: DirectoryIndex index.html *.php
> ```

```
>systemctl restart httpd
```

Volvemos a intentarlo. Ahora nos da HTTP error 500. Podemos inspeccionar con F12: el servidor ha tenido un error.

```
>cd /var/www/html
>php index.php      //ERROR: no conoce a mysqli_connect

>sudo dnf install php-mysqlnd

>php index.php      //Ahora sí podemos acceder desde el server
```

Podemos acceder desde el server, pero desde el navegador externo nos encontramos otro problema, ahora de permisos mysql.

```
>sudo less /var/log/audit/audit.log     //avc denied... httpd no tiene permiso
para conectarse con mysql
>getsebool -a | grep httpd      //httpd_can_network_connect_db está a OFF
>sudo setsebool -P httpd_can_network_connect_db=on
>getsebool -a | grep httpd
```

Ahora sí, si volvemos a actualizar el navegador tendremos éxito - Hemos configurado la pila LAMP.

Vídeo 5 - P2L2 Control de versiones local con git

Suponemos que git está instalado en el sistema

```
>git init mirepo
>ls -la
>cd .git
>ls -la

Configurar nuestro usuario ------------------------------------------------------
>git-config
ó
```

```
>vi ~/.gitconfig

Staging ----------------------------------------------------------------
>touch libro.txt
>git status         //vemos que libro.txt no tiene seguimiento
>git add libro.txt
>git status         //la caja de cereales está en el carrito pero aún no pasamos
por caja
>git commit -m "crear libro"
>git status
>git log            //bitácora de commits
>git log graph
>gitk               //UI
    >>>bg para que siga en el background ejecutando

>cat > passwords    //escribir lo que sea, "titulo"
>git status
>vi .gitignore      //passwords, *.o, imgs
>git status         //passwords ya no aparece, pero sí gitignore
    >git add gitignore
    ó
    >vi .gitignore //.gitignore, que se ignore a sí mismo
>git status

>vi libro.txt       //escribir lo que sea, "subtitulo"
>git diff           //diferencia working directory y stage
>git add libro.txt
>vi libro.txt       //escribir lo que sea
>git diff HEAD      //para comparar con el repositorio (último commit)
>git diff --staged  //para comparar con lo que está en seguimiento
>git add libro.txt
>git diff           //no sale nada, pq todos los cambios están ya staged
>git diff HEAD      //nos salen los dos cambios hechos, pq no están commiteados

>git add libro.txt
>git commit -m "mod libro.txt"

Ramas ------------------------------------------------------------------
>git checkout -b cap1   //crear e ir a rama
>git branch             //lista de ramas

    >>>En gitk vamos a crear una nueva vista para ver las ramas:
        Vista>All branches
        Archivo>Reload
>vi libro.txt           //escribir, "cap1"
>git commit -am "contenido..."

>git checkout main      //nada de lo que hemos añadido en la otra rama se ve
>vi libro.txt           //"titulo nuevo"
>git commit -m "cambiado titulo"

>git checkout cap1
>git log --graph        //las ramas empiezan a diverger... vamos a unirlas

>git merge
```

```
    1.git checkout main
    2.git merge cap1
    3.git log --graph
ó
>git rebase             //desaconsejado --- Nota: pasos A-F están ahí para
recrear el estado que tenía la rama cap1, no por nada más
    A.git checkout -b cap2
    B.vi libro.txt      //"cap2"
    C.git commit -am "añadido cap2"
    D.git checkout main
    E.vi libro.txt      //"subtitulo nuevo"
    F.git commit -m "cambiado subtitulo"

    1.git checkout cap2
    2.git rebase main
    3.git checkout main
    4.git merge cap2
    5.git log --graph   //no se ve la divergencia de ramas como se veía en el
merge de cap1 - es una sola línea

Borrar rama:
>git branch -D cap1     //D mayus para forzar el borrado, minus te avisa de que
hay cambios y tal sin guardar

Deshacer cambios ------------------------------------------------------------
Restore: cambios en el pasado, sobre el working directory
>git restore --source=[id commit, lo tomamos de gitk] libro.txt
>git restore libro.txt  //al no poner commit, asume el último

Revert: para dejar constancia de la vuelta atrás
>git revert [id commit] //hay conflicto, solucionamos con vi las partes
problemáticas
>git add libro.txt
>git revert --continue  //podemos escribir un msj de descripción, cuenta como
commit

Reset: eliminar la historia
>git reset --hard [id commit]   //se elimina el contenido y el commit
```

Diapositivas: Copias de seguridad y control de versiones

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

Copias de seguridad: backup

> Definición aproximada y existencia: el backup que no se comprueba es como si no existiera

Tipos

- Según cantidad de información
    - Completo (Full) vs Selectivo (Partial)
    - Incremental: almacena los no incluidos en el FB (sean del FB o no)

    - Diferencial: almacena respecto a los cambios anteriores (respecto del FB)
- Según estado de la máquina
    - Frío vs Caliente (Máquina apagada vs Máquina operando)
- Según ubicación
    - Local vs Remoto : cuidar espacio y comunicaciones seguras...

Herramientas de copias de seguridad

- Copia binaria – http://www.thegeekstuff.com/2010/10/dd-command-examples/
    - `dd` - Útil para copiar bit a bit el contenido
    - Ejemplos:

```
Copia a otro dispositivo:
>dd if=/dev/sda of=/dev/sdb

Crea una imagen de sda:
>dd if=/dev/sda of=~/hdadisk.img
    Para recuperarla:
    >dd if=hdadisk.img of=/dev/sdb
    Se pueden especificar particiones: sda1, sda2
```

- Copiar archivos y empaquetar
    - `cp`
    - `cpio` - Copia archivos a y desde (archivos)

```
>ls | cpio -ov > /tmp/object.cpio

>cpio -idv < /tmp/object.cpio
```

    - `tar` - https://help.ubuntu.com/community/BackupYourSystem/TAR

```
>tar cvzf MyImages-14-09-17.tar.gz /home/MyImages
>tar -xvf public_html-14-09-17.tar
```

- Sincronizar – https://help.ubuntu.com/community/BackupYourSystem/TAR
    - `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
    - https://rsync.samba.org/examples.html)

```
rsync -a --delete source/ destination/
```

- 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 (vebose)
  - -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

- `rsnapshot`

  Rsnapshot utiliza rsync (y SSH) para automatizar tareas de backup
  - Escrito en perl (sin dependencias externas)

  script vs. Rsnapshot…
  - http://www.mikerubel.org/computers/rsync_snapshots/
  - rsnapshot HOWTO: http://rsnapshot.org/rsnapshot/docs/docbook/rest.html

  Guía de uso: https://docs.rockylinux.org/guides/backup/rsnapshot_backup/
    - Configurar el archivo de configuración con los parámetros (
    - Automatizar su ejecución con systemctl (P3)

- Instantáneas – http://www.tldp.org/HOWTO/LVM-HOWTO/snapshots_backup.html
  - LVM snapshots
- Otros sistemas:
  - p.ej. AMANDA, Bacula, C-Panel, Plesk, etc.
  - Reflexión: ¿Copia vs. Backup? http://www.backupcentral.com/

Otros aspectos a considerar

- Legislación → LOPD y GPDR vs copias forever
- Documentar lo que hacemos
- Tests, Monitorización (P3...), Comprobación
- Automatización
- KISS - Keep It Simple, Stupid

En definitiva, lo que comentaremos en el seminario sobre filosofía de trabajo

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 #

Hay algunas herramientas que controlan /etc directamente (/etc/keeper).

Usa un sistema de control de versiones por debajo (git)

- Podemos incluir archivos de históricos, logs, etc.
- Scripts de monitorización, configuración, tests, etc.
- Son todo ventajas

Git

[TODO LO QUE EXPLICA AQUÍ TAMBIÉN ESTÁ EN EL VÍDEO 5 ^^^ Y SON LAS 4, A MIMIR]