Practica 2 – Instalación y configuración de servicios
Sesión 1

Términos

Superusuario en Ubuntu: **sudo –i**
Ubuntu tiene ip: 192.168.56.105
CentOS tiene ip: 192.168.56.110

Gestores de paquetes:

- Ubuntu: apt-get
- CentOS: yum

DNF y snap pretendían unir los gestores de paquetes en uno, pero al final nada (**NO importante**)

Un **puerto** es como el nº de la puerta de entrada, siendo el número de la dirección la IP

ssh (Secure SHell): Es un protocolo de comunicación encriptado sobre un canal no seguro, permitiendo trabajar en modo remoto. Está basado en la arquitectura cliente-servidor (tanto cliente como servidor son ssh). *Por defecto* trabaja en el puerto 22.

Proceso Ubuntu

<u>Instalación de openssh-server</u>

- **tasksel** (Entrar en modo gráfico para instalación de paquetes)
    - Marcar OpenSsh-Server

  ó
- **apt search openssh** (realmente para nada, solo para comprobar la versión)
- **apt install openssh-server**
    - Se puede comprobar con **ps –AF| grep ssh**
    - También con **ssh ive@localhost** (Haciendo conexión a uno mismo)
        - El fingerprint es un código hash que identifica unívocamente al cliente
        - Puede que no funcione si pones **ssh localhost** siendo root, puesto que estás accediendo al usuario root.
        - **Con know-host, borrar fingerprint**

<u>Configurar archivo ssh para bloquear el acceso desde root</u>

Porque es peligroso que exista esa opción (el root tiene acceso a todo).

- **sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.backup**
- **sudo vi /etc/ssh/sshd_config**
    - PermitRootLogin no (estaba en prohibit-password)

- Restaurar servicio
  systemctl restart sshd

  Desde CentOS: **ssh 192.168.56.105 –l root** ó **ssh 192.168.56.105** no está permitido, comprobando que funciona correctamente

<u>Cambiar puerto, para hacer más difícil el acceso</u>

- **sudo vi /etc/ssh/sshd_config**
    - Port 22022
- **systemctl restart sshd**

  Ahora se debe acceder con **ssh 192.168.56.105 –p 22022 –l ive**

Acceso sin contraseña

Dar llave pública al servidor. En **CentOS**, el "candado" (aka clave pública) está en **id_rsa.pub**, **id_rsa** es la privada

- Generar llaves (**DESDE CENTOS**)
  ssh-keygen
  - **NO SE DEBE ESTAR COMO ROOT (o sí, simplemente cambiar la ruta de almacenamiento)**
  - Almacenar en /home/ive/.ssh/id_rsa – O dejar en blanco ¿?
  - Se debería poner una contraseña, pero la pedirá cada vez que se use (y para este caso no queremos)
    Con **ls –la** se comprueba (se genera carpeta .ssh)

- Copiar al servidor la clave pública (**DESDE CENTOS**)
  ssh-copy-id 192.168.56.105 –p 22022
  (Por defecto te manda todas las claves públicas, las privadas no se pueden enviar (no tiene sentido))

- Comprobar que funciona (**DESDE CENTOS**)
  ssh 192.168.56.105 –p 22022 (en root **–l ive**) (Debería entrar automáticamente)

Proceso CentOS

Lo mismo del otro día, pero con CentOS

- Comprobar si funciona ssh
  systemctl status sshd
- Comprobar si funciona la red
  ip addr
- Si da problemas levantarla
  ifup enp0s3
- Si sigue dando problemas borrar fingerprint
  cd .ssh
  vi known_host
  Borrar ip: Ubuntu …110

  systemctl restart sshd

Bloquear root

En **/etc/ssh/sshd_config**
PermitRootLogin no, **y quitar almohadilla (estaba a yes)**
systemctl restart sshd

Cambiar puerto

Sed (stream editor) para reemplazar en un solo comando. Sed es un editor de textos más potente que vi
sed –e "s/#Port 22/Port 22022/" –i /etc/ssh/sshd_config

Si se prueba da error, se debería leer el fichero de configuración. Indica que hay que decirle a SElinux que se ha cambiado el puerto

- Comprobar interfaz de red levantada
- Levantarla
  ifup enp0s3
- Instalar semanage
  yum search semanage
  yum install policycoreutils-python -y (nombre inacabado, pero es suficiente)

- Listar los puertos
  semanage port –l
- Ver en cuáles de ellos escucha ssh
  semanage port –l | grep ssh
- Avisar del cambio de puerto
  semanage port –a –t ssh_port_t –p tcp 22022
- **semanage port –l | grep ssh** aparece también 22022, le ha indicado que peticiones ssh pueden ir por también por ahí
- **systemctl restart sshd**

ssh ive@192.168.56.110 **–v –p 22022** rechaza la conexión, la culpa es del cortafuegos

- Añadir una excepción al firewall (inmediatamente)
  firewall-cmd --add-port=22022/tcp

ssh ive@192.168.56.110 **–v –p 22022** funciona, pero no es permanente (al apagar se pierde)

- Excepción permanente
  firewall-cmd --add-port=22022/tcp --permanent

  Tampoco funciona (es al reiniciar)
  Se le puede decir manualmente (o haciéndolo sin permanent y luego con permanent)
- **firewall-cmd --reload**

En Ubuntu los comandos del cortafuegos van con **ufw**
ufw enable
ufw status

Si se activa ya, al igual que CentOS, no dejaría ssh
ufw allow 22022 (para hacer una regla permitiendo un puerto)

El acceso sin contraseña está permitido por defecto

Sesión 2

Las copias de seguridad (backups) son necesarias (recordar RAIDs). Se diferencia de la copia normal (cp) en los metadatos adicionales que guarda la de seguridad, por ejemplo, le fecha.

Cosas a tener en cuenta:
- Espacio que ocupa el backup, se debería comprimir para reducirlo
- Tiempo de creación y de restauración (disponibilidad de la copia)
- Atomicidad de las operaciones, que nada cambie mientras se hace la copia
- Seguridad en la copia

Formas
Copia binaria con **dd**, a nivel de bits (replicaciones). Ej: **dd if=/ of=/**
Copia de archivos con **cp**, o **cpio** (en desuso, se usaba principalmente para copias en cinta)
Empaquetamiento con **tar**, que también permite copia y compresión

Sincronización con **rsnapshot**, o **rsync. rsync** utiliza diff para ver las diferencias entre los directorios local y remoto, y solo modifica (en el remoto) las diferencias. Con --delete obligas a reemplazar en el remoto (eliminar lo que no pertenece al local). -aviz (a (de all) combina muchas cosas, v de verbose, z para compresión, i para realizar un informe al final). Cambiando fuente y destino haces restauración. También permite la sincronización a través de SSH, agrupación en un único archivo
Instantáneas (snapshots) con **lvm**

Existen otros métodos más profesionales que incluyen el uso de SW como AMANDA, Bacula, C-Panel, Plesk…

Control de cambios
Lo básico es realizar copias de archivos antes de modificarlo. También hay herramientas que ayudan a mantener la configuración (/etc/keeper)

Git
Git es un sistema de control de versiones que entre otras cosas permite el seguimiento de cambios y la creación de distintas ramas de trabajo. En CentOS no viene por defecto. Para instalarlo: yum install git

Su funcionamiento se divide en 4 zonas, las 3 primeras locales:
- Directorio de trabajo donde se realizan modificaciones
- Stage, zona donde se incluyen y registran los cambios que se realizan
- Commit, zona donde los cambios se vuelven permanentes
- Repository, zona remota donde se almacena la aplicación

Cada vez que se realiza un commit se le genera un hash (llamado SHA1) que lo identifica. Estos commits se van apilando, siendo HEAD el último (e incluye todo).
A parte de los commits existen otros dos tipos de objetos: **blobs** (archivos, con sus inodos y datos) y **trees** (directorios), también identificados unívocamente

Instrucciones
Para comprobación y recogida de información: **git log** y **git status**
Para iniciar el repositorio: **git clone** (copia remota) y **git init**
Para configurar: **git config**
Añadir archivos al seguimiento: **git add.** Con el archivo **.gitignore** indicamos qué archivos no se quieren seguir
Realización de commits: **git commit** (con --amend se modifica el mensaje)
Control de modificaciones: **git diff.** Permite ver qué diferencias hay entre lo que se ha modificado y lo que está siendo seguido en el stage. Si primero se modifica y luego se añade no se mostrará nada. (**diff HEAD** para ver diferencias con el repositorio, **diff --staged** para ver diferencias entre el stage y el repositorio)
Restauración: **git reset** para actualizar el HEAD a un commit concreto. **git revert** para desaplicar los cambios hechos en un commit (creando a su vez un nuevo commit, -n para evitarlo)
Creación de ramas: **git branch**
Cambio de ramas: **git checkout** (-b para crearla)
Salvaguardado: **git stash**, salva el directorio actual sin tener que hacer un commit, permitiendo cambiar de rama y dejar el trabajo a medias. **git stash apply** para aplicar los cambios
Unión de ramas: **git merge**
Envío a remoto: **git push**
Traer cambios desde remoto: **git pull**

Si trabajamos usando SSH, todos los usuarios deben tener acceso a la máquina, o se crea un usuario git compartido (evitando que el usuario git tenga acceso a una Shell)

Sesión 3

Instalación de un servidor web básico, en nuestro caso, LAMP (Linux Apache MariaDB Python/PHP - SO, software que permite dar el servicio web, almacenamiento, lenguaje e intérprete de órdenes)

Se elige un lenguaje interpretado por velocidad, por no perder tiempo compilando (además las modificaciones se ven inmediatamente).
Se podría tener más de una LAMP (o sistema similar) en una misma máquina.

Servidor especial que se comunica a través de internet (importante su software que proporciona servicio)
HTML (HyperText Markup Language): Documentos con enlaces a otros documentos o a sí mismo (Hypertext)
HTTP (HyperText Transfer Protocol): Protocolo de transferencia de archivos HTML
URI (Uniform Resource Identifier) y URL (Uniform Resource Locator)

Proceso

sudo –i

Instalación
Ubuntu

- tasksel (LAMP server) (hace falta actualizar antes) (se inicializa automáticamente)
- Se ha instalado Apache, MariaDB (MySQL) y PHP. Comprobar:
    - systemctl status apache2
    - php-a
    - mysql –u root –p (show databases),

CentOS

- ifup enp0s3
- yum install httpd –y
    - Comprobar: systemctl status httpd (dice disabled (no se ejecuta al reiniciar la máquina) and inactive)
    - **systemctl enable httpd**
    - **systemctl start httpd**
- yum install php
    - php –a
- yum install mariadb-server –y (sin server es el cliente)
    - systemctl status mariadb
        - systemctl enable mariadb
        - systemctl start mariadb
    - mysql_secure_installation (poner contraseña para el root)
        - yes remove anonymous
        - yes disallow root login
        - yes remove test
        - yes reload privilege
    - mysql –u root –p
        - create database mi_db;
        - show databases;

Se ha modificado el script:

```php
<?php
$enlace = mysqli_connect("localhost", "root", "practicas,ISE", "mi_db");

if (!$enlace) {
    echo "Error: No se pudo conectar a MySQL." . PHP_EOL;
    echo "errno de depuración: " . mysqli_connect_errno() . PHP_EOL;
    echo "error de depuración: " . mysqli_connect_error() . PHP_EOL;
    exit;
}

echo "Éxito: Se realizó una conexión apropiada a MySQL! La base de datos mi_bd es gen
ial." . PHP_EOL;
echo "Información del host: " . mysqli_get_host_info($enlace) . PHP_EOL;

mysqli_close($enlace);
?>
```

Despues de instalar SSH server se enciende automáticamente en CentOS en Ubuntu no.

Añadirlo: vi /var/www/html/miscript.php

CentOS

- Problema con el firewall, permitir puerto:
  - firewall-cmd --add-port=80/tcp --permanent
  - firewall-cmd --reload
- Problema con intérprete, no se ejecuta el script, editar /etc/httpd/conf/httpd.conf (apache)
  - Añadir a: DirectoryIndex index.html *.php
  - systemctl restart httpd
- Error con intérprete (otra vez), instalar modulo php con mysql
  - yum install php-mysql –y
- Problema con SELinux
  - **getsebool –a | grep hhtpd** nos dice variables de configuración (flags)
  - setsebool –P httpd_can_network_connect_db on

<u>Fail2ban</u>

fail2ban usa el concepto de jail, no significa que no puedan acceder, solo los destaca
Se puede configurar para que los que estén en la cárcel pasen a banned

CentOS

- yum install epel-release –y + yum install fail2ban
- systemctl enable fail2ban + systemctl start fail2ban
- fail2ban-client status,
- Configuración en /etc/fail2ban/jail.conf
  - cp jail.conf jail.local -a
  - Modificar .local
  - Descomentar [sshd] y la de debajo (**NO PUEDE HABER ESPACIOS AL PRINCIPIO**)
- Si se falla 5 veces en acceder con ssh, en **fail2ban-client status sshd** se ve la IP

Esto por ahora es solo una lista negra

- En JAILS, sshd, cambiar port a 22022
- systemclt restart fail2ban

Ya si estarían baneados, para quitar el ban: **fail2ban-client set sshd unbanip 192.168.56.106**

Root Kit Hunter (rkhunter)
Similar a un antivirus. Se instala como siempre, y genera informes.

Screen
Entrando desde ssh
Interesante cuando se trabaja con cluster u operaciones con múltiples procesos.
Para recuperar en caso de cierre inesperado

- yum install screen –y

El comando genera una especie de nueva terminal, por lo que si se cierra la otra terminal inesperadamente con **screen –list** se pueden ver las ventanas (con su ID), y con **screen –r <ID Proceso>** se restaura

Con CTRL + A + D se desvincula, se sale de la screen pero no se cierra.

Comandos	
Sesión 1 – Ubuntu	
tasksel	Modo gráfico para instalación de paquetes
apt install openssh-server	Instala OpenSSH
ps –AF | grep ssh	Comprueba si ssh está en funcionamiento
ssh ive@localhost	Conexión a uno mismo
/etc/ssh/sshd_config PermitRootLogin no Port 22022	 Bloquea el acceso a root desde ssh Cambia el puerto a 22022
systemctl restart sshd	
ssh <IP> –l <usuario> -p <puerto>	
ssh-keygen	Generación de clave pública y privada en directorio actual (hacer desde CentOS)
ssh-copy-id <IP> -p <puerto>	Copiar clave pública hacia IP (desde CentOS)
ufw enable	Activar firewall
ufw status	Estado del firewall
ufw allow 22022	Hacer regla permitiendo puerto
Sesión 1 – CentOS	
/etc/ssh/sshd_config (quitar almohadillas) PermitRootLogin no	
sed –e “s/#Port 22/Port 22022/” –i /etc/ssh/sshd_config	Editar archivo en un commando (reemplazando)
ifup enp0s3 ifup enp0s8	Levantar red Internet Levantar red con Host
yum install policycoreutils-python	Instalar semanage
semanage port –l | grep ssh	Listar puertos de ssh
semanage port –a –t ssh_port_t –p tcp 22022	Avisar de cambio Puerto a 22022
firewall-cmd --add-port=22022/tcp --permanent	Añadir excepción a firewall inmediatamente Permanentemente
firewall-cmd --reload	Reiniciar firewall
Sesión 3 – Ubuntu	
tasksel – LAMP server	
systemctl status apache2 php –a	Comprobación funcionamiento LAMP

mysql –u root –p (show databases)	
Sesión 3 – CentOS	
yum install httpd yum install php yum install mariadb-server	Instalación LAMP
mysql_secure_installation (yes a todo)	Terminar configuración MariaDB
systemctl status httpd php –a systemctl status mariadb	
/var/www/html	Directorio donde añadir archivos al servidor
firewall-cmd --add-port=80/tcp --permanent	Permitir conexión MariaDB
/etc/httpd/conf/httpd.conf DirectoryIndex index.html *.php	 Permite ejecución de scripts php
yum install php-mysql	Módulo php con mysql
getsebool –a | grep httpd	Flags de httpd
setsebool –P httpd_can_network_connect_db on	Permitir conexión httpd a base de datos
Fail2ban	
yum install epel-release yum install fail2ban	
fail2ban-client status	Estado de las cárceles, añadir (ssh) para ver las IPs del servicio
/etc/fail2ban/jail.conf -> cp a jail.local Descomentar [sshd] y línea de debajo Descomentar [JAILS], añadir port 22022	 Activar fail2ban para sshd Activa el modo de baneos
fail2ban-client set sshd unbanip <IP>	Desbanear IP
Screen	
yum install screen	
screen –list	Lista screens abiertas
screen –r <ID>	Restaura screen
CTRL + A + D	Desvincula screen sin cerrarla

Tipo de preguntas de examen:

- **¿Qué hace git diff?** Control de modificaciones. Sin parámetros permite ver qué diferencias hay entre lo que se ha modificado y lo que está siendo seguido en el stage.
- **¿CentOS por defecto usa MariaDB?** No, hay que instalarlo
- **Pasos para usar ssh en CentOS:** Bloquear root y cambiar puerto (cambiar configuración ssh, informar SELinux, modificar firewall para permitir el uso del puerto)
- **Comandos para firewall en UbuntuServer:** ufw
- **Diferencias entre /* y /. en rysnc:** Todos los archivos sin incluir o no los ocultos
- **Como crear archivo comprimido a partir de una carpeta usando tar:** tar cvzf archivo.tar.gz /home
- **¿Puede variar el contenido del directorio y del repositorio?** Por supuesto, al repositorio solo van los cambios que tenemos en nuestro directorio y se han indicado en un commit.
- **Diferencia entre /. y no poner nada en rsync:** No coge los ocultos (sin / coge el directorio también)
- **Explicar las opciones más habituales de rsync:** Con --delete obligas a reemplazar en el remoto (eliminar lo que no pertenece al local). -aviz (a (de all) combina muchas cosas, v de verbose, z para compresión, i para realizar un informe al final).
- **Tipo de modelo git usado en clase**
    - Distribuido

    - Desarrollo
    - DDG
    - Otro
- **Cómo viene de serie el firewall de Ubuntu:** Desactivado
- **Problemas al usar un archivo PHP en httpd y MariaDB:** En primera instancia hay que indicar a hhtpd que lea los archivos con extensión .php. Luego hay que instalar el paquete PHP que funciona con MySQL, pues no viene por defecto. (Tambíen hay que añadir puerto 80 al firewall y cambiar flag de SELinux)
- **Cómo activar un puerto usando semanage:** Instalar el paquete necesario y notificar con: semanage port –a –t ssh_port_t –p tcp 22022
- **Enumerar los pasos para activar un puerto en CentOS:** Modificar archivo de configuración ssh, usar semanage para indicar a SELinux del cambio, crear excepción permanente en el firewall para permitir el puerto.
- **Significado de rkhunter:** RootKitHunter