

**INGENIERÍA DE SERVIDORES (2016-2017)**  
GRADO EN INGENIERÍA INFORMÁTICA  
UNIVERSIDAD DE GRANADA

---

## Memoria Práctica 2

---

Adrián Morente Gabaldón

25 de noviembre de 2016

## Índice

1	a) Liste los argumentos de yum necesarios para instalar, buscar y eliminar paquetes. b) ¿Qué ha de hacer para que yum pueda tener acceso a Internet en el PC del aula? (Pistas: archivo de configuración en /etc, proxy: starga-te.ugr.es:3128) c) ¿Cómo añadimos un nuevo repositorio?	6
1.1	Argumentos para uso de yum . . . . .	6
1.2	Acceso a Internet en el PC del aula . . . . .	6
1.3	Añadir repositorios a yum . . . . .	6
2	a) Liste los argumentos de APT necesarios para instalar, buscar y eliminar paquetes. b) ¿Qué ha de hacer para que APT pueda tener acceso a Internet en el PC del aula? c) ¿Cómo añadimos un nuevo repositorio?	7
2.1	Argumentos para uso de APT . . . . .	7
2.2	Acceso a Internet en el PC del aula . . . . .	7
2.3	Añadir repositorios a APT . . . . .	7
3	a) ¿Con qué comando puede abrir/cerrar un puerto usando ufw? Muestre un ejemplo de cómo lo ha hecho. b) ¿Con qué comando puede abrir/cerrar un puerto usando firewall-cmd en CentOS? Muestre un ejemplo de cómo lo ha hecho. c) Utilice el comando nmap para ver que, efectivamente, los puertos están accesibles.	8
3.1	Comandos de apertura/cerrado de puertos con ufw . . . . .	8
3.2	Comandos de apertura/cerrado de puertos con firewall-cmd . . . . .	8
3.3	Uso de nmap para listar puertos abiertos . . . . .	9
4	¿Qué diferencia hay entre telnet y SSH?	10
5	a) ¿Para qué sirve la opción -X? b) Ejecute remotamente, es decir, desde la máquina anfitriona (si tiene Linux) o desde la otra máquina virtual, el comando gedit en una sesión abierta con SSH. ¿Qué ocurre?	10
5.1	Opción -X de SSH . . . . .	10
5.2	Ejecución remota con interfaz gráfica mediante SSH . . . . .	11
6	Muestre la secuencia de comandos y las modificaciones a los archivos correspondientes para permitir acceder a la consola remota sin introducir la contraseña. Pruebe que funciona. (Pistas: ssh-keygen, ssh-copy-id).	12
7	¿Qué archivo es el que contiene la configuración del servicio SSH? ¿Qué parámetro hay que modificar para evitar que el usuario root acceda? Cambie el puerto por defecto y compruebe que puede acceder.	14
8	Indique si es necesario reiniciar el servicio ¿Cómo se reinicia un servicio en Ubuntu? ¿y en CentOS? Muestre la secuencia de comandos para hacerlo.	15

9 Muestre los comandos que ha utilizado en Ubuntu Server y en CentOS (aunque en este último puede utilizar la GUI, en tal caso, realice capturas de pantalla). Compruebe que la instalación ha sido correcta.	16
10 Realice la instalación usando GUI o PowerShell y compruebe que el servicio está funcionando accediendo a la MV a través de la anfitriona.	18
11 Muestre un ejemplo de uso del comando patch (p.ej. un programas sencillo HolaMundo con algún fallo concreto [bucle infinito, por ejemplo], y otro programa con dicho fallo arreglado)	19
12 Realice la instalación de esta aplicación (Webmin) y pruebe a modificar algún parámetro de algún servicio. Muestre las capturas de pantalla pertinentes así como el proceso de instalación.	21
13 Instale phpMyAdmin, indique cómo lo ha realizado y muestre algunas capturas de pantalla. Configure PHP para poder importar BDs de hasta 25MiB (en vez de los 8 MiB de límite por defecto). Indique cómo ha realizado el proceso y muestre capturas de pantalla.	23
14 Visite al menos una de las webs de los software mencionados y pruebe las demos que ofrecen realizando capturas de pantalla y comentando qué está realizando.	25
15 a) Ejecute los ejemplos de find, grep b) Escriba el script que haga uso de sed para cambiar la configuración de SSH y reiniciar el servicio. c) Muestre un ejemplo de uso para awk.	27
15.1 Ejemplos de find y grep . . . . .	27
15.2 Script de sed para modificar SSH . . . . .	28
15.3 Ejemplo de uso de awk . . . . .	29
16 Escriba el script para cambiar el acceso a SSH usando PHP o Python.	30
17 Abra una consola de Powershell y pruebe a parar un programa en ejecución, realice capturas de pantalla y comente lo que muestra.	31

## Índice de figuras

3.1. Cierre y reapertura de puertos para SSH y HTTP. - Adrián Morente Gabaldón [20/11/2016] . . . . .	8
3.2. Apertura de puertos, con sus correspondientes comprobaciones. - Adrián Morente Gabaldón [20/11/2016] . . . . .	9
3.3. Visualización de puertos abiertos del sistema. - Adrián Morente Gabaldón [20/11/2016]	10
5.1. Ejecución de gedit con interfaz gráfica de forma remota a través de SSH. - Adrián Morente Gabaldón [21/11/2016] . . . . .	12

6.1. Generación de clave privada y acceso de forma remota a través de SSH. - Adrián Morente Gabaldón [21/11/2016] . . . . .	13
7.1. Acceso correcto con usuario normal, pero acceso de forma errónea como usuario root. - Adrián Morente Gabaldón [23/11/2016] . . . . .	15
8.1. Reinicio de servicios en Ubuntu Server. - Adrián Morente Gabaldón [20/11/2016]	15
8.2. Reinicio de servicios en CentOS. - Adrián Morente Gabaldón [22/11/2016] . . .	16
9.1. Comprobación de la instalación del servicio Apache en Ubuntu Server. - Adrián Morente Gabaldón [22/11/2016] . . . . .	17
9.2. Comprobación de la instalación del servicio Apache en CentOS. - Adrián Morente Gabaldón [22/11/2016] . . . . .	18
10.1. Página de demostración del servicio IIS. - Adrián Morente Gabaldón [22/11/2016]	19
11.1. Ejecución de diff-patch con dos programas de prueba. - Adrián Morente Gabaldón [22/11/2016] . . . . .	20
12.1. Demostración del funcionamiento de Webmin. - Adrián Morente Gabaldón [24/11/2016]	22
12.2. Configuración de bloqueo del acceso al servicio para ciertas direcciones IP. - Adrián Morente Gabaldón [24/11/2016] . . . . .	23
13.1. El instalador de phpMyAdmin nos dará a elegir el servidor web que queremos utilizar junto con dicho administrador. - Adrián Morente Gabaldón [23/11/2016]	24
13.2. Pantalla de inicio del servicio PHPMyAdmin. - Adrián Morente Gabaldón [23/11/2016]	25
13.3. Configuración del límite de tamaño de bases de datos en PHPMyAdmin. - Adrián Morente Gabaldón [22/11/2016] . . . . .	25
14.1. Si miramos la barra de menú, nos encontramos con varias pestañas, cada una de las cuales nos permite gestionar una parte del servidor: podemos añadir clientes, traductores DNS, dominios de correo electrónico, cambiar datos como contraseña y lenguaje, etc. - Adrián Morente Gabaldón [22/11/2016] . . . .	26
14.2. En la pestaña Monitor encontramos la lista de servidores administrados y el periodo con el que queremos refrescar las estadísticas del profiler usado. En este caso solo tendremos un servidor de prueba ya que, como sabemos, se trata de una demostración gratuita de un servicio de pago. Sin embargo, es ilustrativo en cuanto a las opciones que podemos usar; si miramos el margen izquierdo, encontramos las opciones de monitorización de procesador, CPU, estado de una configuración de discos en RAID, etc. En este caso no podremos cambiar nada por la naturaleza ficticia de la página. - Adrián Morente Gabaldón [22/11/2016]	27
15.1. El comando <i>ps</i> nos muestra todos los procesos que hay en ejecución en el sistema, que gracias a <i>grep</i> filtramos fácilmente. Así nos quedamos con el usuario que está ejecutando el proceso, la hora y el tiempo de ejecución, el directorio y el nombre del proceso, entre otras cosas. - Adrián Morente Gabaldón [23/11/2016]	27
15.2. Empezamos listando el contenido de nuestro directorio /home y de las carpetas involucradas (/docs y /PDFs). A continuación, ejecutamos el comando de prueba y vemos como <i>find</i> busca los archivos que cumplen el requisito (en este caso, que su nombre termine en <i>pdf</i> ) y emprende con ellos la acción determinada: copiarlos a la carpeta /PDFs. Una vez hecho, listamos el contenido de esta carpeta para demostrar su efectividad y sencillez de uso. - Adrián Morente Gabaldón [23/11/2016] . . . . .	28

15.3. Para empezar, imprimos el contenido del archivo de configuración para ver su contenido (puerto 22). A continuación, vemos una ejecución del script errónea, ya que intentamos acceder a un puerto reservado. Justo después, le ponemos un puerto correcto y volvemos a imprimirlo para demostrar su funcionamiento. - Adrián Morente Gabaldón [24/11/2016] . . . . .	29
15.4. Como ya vimos en la pequeña descripción del código, la ejecución de este mini-programa imprime el puerto 22, que es el definido por defecto para el servicio SSH. - Adrián Morente Gabaldón [24/11/2016] . . . . .	30
16.1. Empezamos imprimiendo una parte del fichero de configuración para comprobar que su puerto está intacto (22). Acto seguido, ejecutamos el script descrito y volvemos a imprimir dicha información, verificando que la modificación se hizo correctamente. Para finalizar, reiniciamos el servicio para que se apliquen los cambios. - Adrián Morente Gabaldón [24/11/2016] . . . . .	31
17.1. Empezamos obteniendo la lista de procesos con el comando <i>Get-Process</i> , y elegimos (por ejemplo) el proceso <i>VBoxTray</i> , el cual se encarga de mantener el icono de las <i>VirtualBox Guest Additions</i> a mano en la barra de tareas de Windows. Con el comando mencionado arriba, fácilmente detenemos dicha aplicación. Volvemos a imprimir los procesos en curso para demostrar su detención. - Adrián Morente Gabaldón [23/11/2016] . . . . .	32

## Índice de tablas

1. a) Liste los argumentos de yum necesarios para instalar, buscar y eliminar paquetes. b) ¿Qué ha de hacer para que yum pueda tener acceso a Internet en el PC del aula? (Pistas: archivo de configuración en /etc, proxy: stargate.ugr.es:3128) c) ¿Cómo añadimos un nuevo repositorio?

### 1.1. Argumentos para uso de yum

Los argumentos necesarios para el uso de yum para instalación, búsqueda y eliminación de paquetes son, respectivamente, los siguientes [3]:

- `sudo yum install -y <nombre>`
- `sudo yum search <nombre>`
- `sudo yum remove <nombre>`

### 1.2. Acceso a Internet en el PC del aula

Como vemos en la documentación de CentOS [2], para utilizar yum con una conexión a Internet mediante proxy, hemos de modificar el archivo de configuración ubicado en `/etc/yum.conf` cumplimentando el siguiente contenido:

```
# The proxy server - URL completa del proxy y número del puerto TCP
proxy = http://stargate.ugr.es:3128
# Detalles de la cuenta de usuario
proxy_username = <usuario>
proxy_password = <contraseña>
```

### 1.3. Añadir repositorios a yum

Los repositorios son el modo de almacenamiento de todas las fuentes y enlaces de donde podemos obtener los paquetes disponibles para su instalación en el sistema. Para añadir repositorios a yum disponemos de varias opciones, las cuales se mantienen todas por temas de retrocompatibilidad con versiones más antiguas del sistema operativo. Esta vez, usaremos la documentación oficial de RedHat para ilustrarnos [1] ya que como sabemos, el código que compone CentOS deriva (gracias a su comunidad) del proyecto RedHat; y que por tanto nos es muy útil en cuanto al aprendizaje de Yum.

- En el archivo ubicado en `/etc/yum.repos.d` hemos de añadir un fichero de texto con extensión `.repo`, cuyo contenido sea la URL del repositorio deseado. A continuación, actualizamos la lista en caché de yum (con `sudo yum update`) y ya lo tenemos a nuestra disposición para su uso.
- La otra opción y más rápida, es ejecutar como usuario el comando `sudo yum-config-manager --add-repo <URL>`.

2. a) Liste los argumentos de APT necesarios para instalar, buscar y eliminar paquetes. b) ¿Qué ha de hacer para que APT pueda tener acceso a Internet en el PC del aula? c) ¿Cómo añadimos un nuevo repositorio?

### 2.1. Argumentos para uso de APT

Los argumentos necesarios para el uso de APT para instalación, búsqueda y eliminación de paquetes son, respectivamente, los siguientes:

- `sudo apt-get install -y <nombre>`
- `sudo apt-cache search <nombre>`
- `sudo apt-get remove <nombre>`

### 2.2. Acceso a Internet en el PC del aula

A diferencia de CentOS, en Ubuntu Server es más sencillo utilizar una proxy para poder conectarnos a Internet a través del PC del aula; tan solo hay que introducir y ejecutar el siguiente comando en la terminal:

```
export HTTP_PROXY=http://stargate.ugr.es:3128
```

### 2.3. Añadir repositorios a APT

Para añadir repositorios a la herramienta APT, tenemos varias alternativas bien conocidas:

- El comando `sudo add-apt-repository ppa:<nombre-repositorio>` es la forma más rápida de incluir un repositorio; acto que debemos seguir con el comando `sudo apt-get update` para actualizar la caché de repositorios de APT y poder empezar a utilizarlos.
- Otra forma es introducir la URL del repositorio deseado en el archivo de configuración de APT `/etc/apt/sources.list`; cosa que podemos hacer tanto a través de un editor de textos, o con el siguiente comando en la terminal:

```
<url> >> /etc/apt/sources.list
```

3. a) ¿Con qué comando puede abrir/cerrar un puerto usando *ufw*? Muestre un ejemplo de cómo lo ha hecho.  
b) ¿Con qué comando puede abrir/cerrar un puerto usando *firewall-cmd* en CentOS? Muestre un ejemplo de cómo lo ha hecho. c) Utilice el comando *nmap* para ver que, efectivamente, los puertos están accesibles.

### 3.1. Comandos de apertura/cerrado de puertos con *ufw*

Como vemos fácilmente en el manual de *ufw* a través de la terminal en Ubuntu, podemos abrir y cerrar puertos con el comando *sudo ufw enable/disable*. Para la muestra, he realizado el cerrado y la consiguiente reapertura de los puertos 22 y 80; que se corresponden con los servicios *SSH* y *http* respectivamente. Otra forma de aceptar el acceso al servicio proporcionado por un puerto, es utilizar *ufw allow/reject <número>[/protocolo]*; de forma que podemos aceptar (o denegar con *reject*) las solicitudes al puerto identificado por *<número>* además de poder proporcionar el tipo de protocolo deseado (TCP o UDP). Si no especificamos este último parámetro, se aceptarán/denegarán las solicitudes de ambos tipos. Esto se documenta en la siguiente imagen:

```
adri@usl4:~$ sudo ufw disable 80
El cortafuegos está detenido y deshabilitado en el arranque del sistema
adri@usl4:~$ sudo ufw enable 80
El cortafuegos está activo y habilitado en el arranque del sistema
adri@usl4:~$ sudo ufw disable 22
El cortafuegos está detenido y deshabilitado en el arranque del sistema
adri@usl4:~$ sudo ufw enable 22
El cortafuegos está activo y habilitado en el arranque del sistema
adri@usl4:~$
adri@usl4:~$ sudo ufw allow 22/tcp
Regla añadida
Regla añadida (v6)
adri@usl4:~$ sudo ufw reject 80/udp
Regla añadida
Regla añadida (v6)
adri@usl4:~$
```

Figura 3.1: Cierre y reapertura de puertos para SSH y HTTP. - Adrián Morente Gabaldón  
[20/11/2016]

### 3.2. Comandos de apertura/cerrado de puertos con *firewall-cmd*

Mientras que en Ubuntu hemos utilizado *ufw* para la administración de puertos, en CentOS utilizamos la herramienta *firewall-cmd*, cuya funcionalidad viene bien explicada en su propio manual, y de cuyas opciones más destacadas hice uso en la prueba adjunta más abajo. Para empezar, comprobamos que el firewall del sistema está activo y en ejecución con el comando *systemctl status firewalld*. En caso de que no lo esté, lo activamos fácilmente con *systemctl enable firewalld*. Aunque en su documentación muestran muchos usos y comandos distintos,



nos restringiremos por ahora a listar los puertos abiertos, y abrir o cerrar algunos; todo esto mediante las siguientes opciones de *firewall-cmd*:

- **firewall-cmd --list-ports**: imprime en pantalla los puertos que hemos abierto manualmente.
- **firewall-cmd --add-port=<número>/<protocolo>**: añade a firewalld (o abre, más bien) el puerto especificado, aceptando las solicitudes que incluyen el protocolo determinado. (Similar a lo mencionado previamente con *ufw* en Ubuntu Server).
- **firewall-cmd --remove-port=<número>/<protocolo>**: elimina de firewalld (o cierra) el puerto especificado para un protocolo dado. Comportamiento similar pero contrario al punto anterior.

```
[amorente@localhost ~]$ systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: active (running) since dom 2016-11-20 14:01:13 CET; 3h 29min ago
     Main PID: 646 (firewalld)
    CGroup: /system.slice/firewalld.service
            └─646 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid

nov 20 14:00:30 localhost.localdomain systemd[1]: Starting firewalld - dyn...
nov 20 14:01:13 localhost.localdomain systemd[1]: Started firewalld - dyna...
Hint: Some lines were ellipsized, use -l to show in full.
[amorente@localhost ~]$ firewall-cmd --list-ports
[amorente@localhost ~]$ firewall-cmd --add-port=80/tcp
success
[amorente@localhost ~]$ firewall-cmd --add-port=22/udp
success
[amorente@localhost ~]$ firewall-cmd --list-ports
80/tcp 22/udp
[amorente@localhost ~]$
```

Figura 3.2: Apertura de puertos, con sus correspondientes comprobaciones. - Adrián Morente Gabaldón [20/11/2016]

### 3.3. Uso de nmap para listar puertos abiertos

Otra forma de comprobar qué servicios estamos ofreciendo de forma activa es mediante el uso de la herramienta *nmap*. Mientras que en Ubuntu Server viene instalada por defecto, en mi caso he tenido que instalarla manualmente en CentOS. Procederemos de la siguiente forma:

```
nmap localhost
```

Esto listará los servicios disponibles, el estado actual (abierto o cerrado) y el puerto en el que se ofrecen. Como información adicional, muestra la versión de Nmap que estamos utilizando, la fecha y hora actuales, y la latencia de conexión al host entre otras cosas. He aquí una muestra:

```
[amorente@localhost ~]$ nmap localhost
Starting Nmap 6.40 ( http://nmap.org ) at 2016-11-20 18:54 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00053s latency).
Other addresses for localhost (not scanned): 127.0.0.1
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
80/tcp    open  http
631/tcp    open  ipp
Nmap done: 1 IP address (1 host up) scanned in 0.09 seconds
[amorente@localhost ~]$
```

Figura 3.3: Visualización de puertos abiertos del sistema. - Adrián Morente Gabaldón  
[20/11/2016]

#### 4. ¿Qué diferencia hay entre telnet y SSH?

Tanto telnet como SSH son dos protocolos de red que permiten acceder de forma remota a otra máquina o servidor para facilitar su administración o gestión. La principal diferencia reside en que telnet es una herramienta más antigua, quizás ya en desuso debido a SSH, que viene a implementar la misma funcionalidad pero de una manera más segura y efectiva. *Secure Shell*, que se trata del significado de las siglas de SSH, viene reemplazando a telnet en prácticamente cualquier lugar, y es la herramienta de gestión remota más usada hoy día.

- Mientras que telnet envía los datos en texto plano, SSH utiliza un encriptado para que solo el destinatario reciba el mensaje.
- Telnet no utiliza ningún tipo de autenticación, lo que permite acceder al mensaje enviado a cualquiera; mientras que SSH utiliza una clave pública de autenticación que mantiene oculto el envío.

Está claro que SSH ofrece una manera más segura de acceder a una máquina remota; la desventaja es que al utilizar un cifrado y una clave pública, se requiere de un mayor ancho de banda para el envío de datos (cosa que hoy en día no notamos, pero que en sus inicios pasó factura con las conexiones lentas).

#### 5. a) ¿Para qué sirve la opción -X? b) Ejecute remotamente, es decir, desde la máquina anfitriona (si tiene Linux) o desde la otra máquina virtual, el comando gedit en una sesión abierta con SSH. ¿Qué ocurre?

##### 5.1. Opción -X de SSH

Como vemos en el manual oficial de SSH (ya sea a través de consola o desde la web de OpenSSH [7]), la opción -X nos permite la ejecución con entorno gráfico de programas en

la máquina remota pero en el núcleo del cliente. Es decir, la máquina servidor tan solo envía el núcleo del programa pero la interfaz gráfica se ejecuta en el entorno del cliente cargada de forma local. Esto supone una gran ventaja ya que libera mucha carga de envío de datos a través de la red.

## 5.2. Ejecución remota con interfaz gráfica mediante SSH

Ya hemos visto el uso de `ssh -X` en el punto anterior, y ahora pasaremos a un caso práctico.

1. Para empezar, tenemos las dos máquinas virtuales de CentOS 7 y Ubuntu Server 14.04 encendidas, ambas con la terminal abierta.
2. A partir del comando `ifconfig` en ambas máquinas, la terminal nos muestra las redes a las que están conectadas cada una de las máquinas, con sus correspondientes direcciones.
3. Una vez que tenemos las direcciones (`eth0` en Ubuntu Server y `enp0s3` en CentOS, en mi caso), probamos a hacer ping entre las máquinas, para verificar que existe una vía de comunicación entre ellas.
4. Hecho esto, pasamos a conectarnos por SSH. Para este supuesto práctico, he probado a conectarme a Ubuntu Server desde CentOS. Para ello, necesitamos ingresar como parámetro a SSH el nombre de usuario con el que queremos iniciar sesión en la máquina remota, junto con la dirección de dicha máquina de la siguiente forma:

```
ssh -X adri@10.0.2.4
```

5. Hecho esto, SSH nos pedirá la contraseña del usuario especificado y al introducirla correctamente ya tendremos acceso. Llegados aquí, con tan solo ejecutar `gedit` vía SSH ya podremos utilizar dicha herramienta con interfaz gráfica.

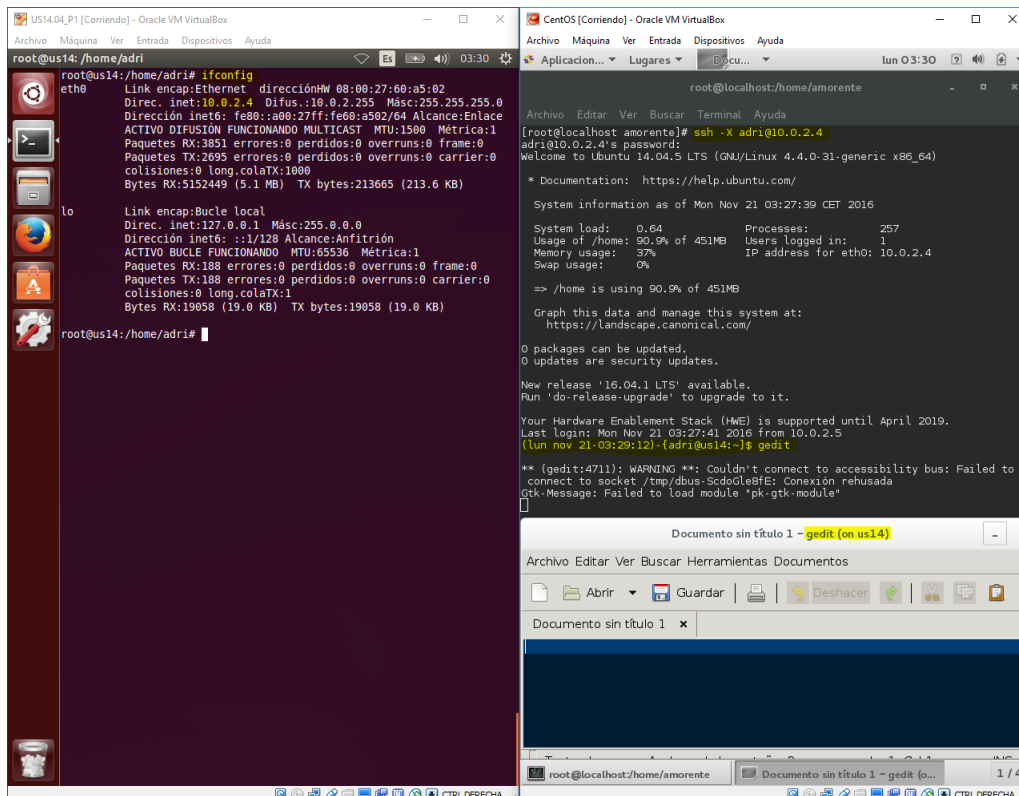


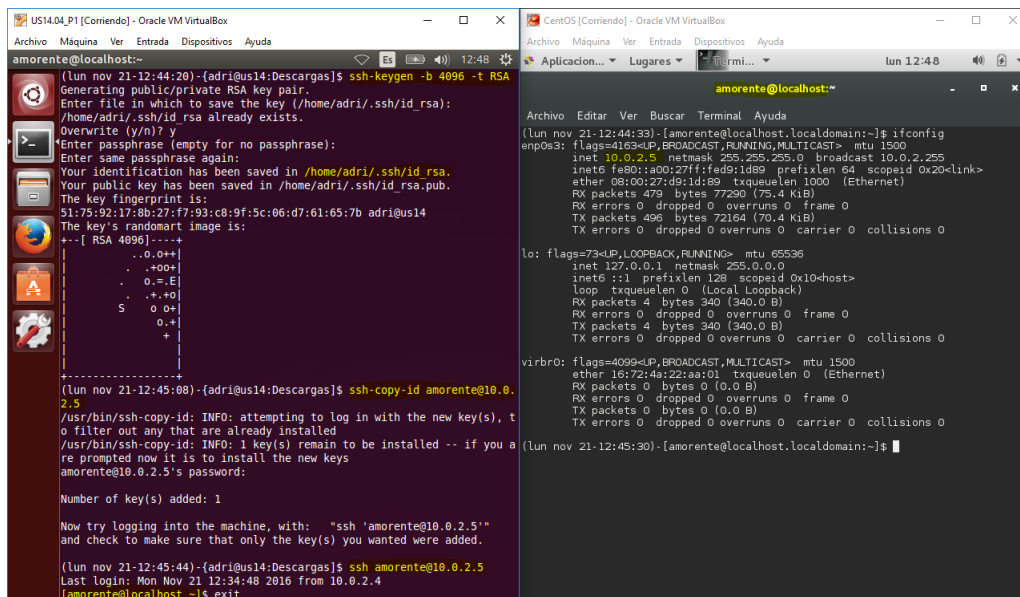
Figura 5.1: Ejecución de gedit con interfaz gráfica de forma remota a través de SSH. - Adrián Morente Gabaldón [21/11/2016]

## 6. Muestre la secuencia de comandos y las modificaciones a los archivos correspondientes para permitir acceder a la consola remota sin introducir la contraseña. Pruebe que funciona. (Pistas: ssh-keygen, ssh-copy-id).

Si vamos a estar accediendo de forma frecuente a una máquina remota mediante Secure Shell, nos puede interesar saltarnos el paso de introducir la contraseña de usuario cada vez que accedemos. Esto se consigue a través de la importación de una clave pública con el sistema algorítmico **RSA**, que se trata del algoritmo más utilizado para cifrado y firma digital. El procedimiento a seguir, realizado desde el directorio */home* del usuario o del administrador raíz, es el siguiente:

1. Para empezar, podemos usar la herramienta *ssh-keygen* para generar una clave dedicada al uso mencionado, y que a través de las opciones *-b* y *-t* nos permite especificar el número de bytes que queremos utilizar en la firma y el algoritmo de cifrado a utilizar, respectivamente.

2. Tras hacer esto, la herramienta nos preguntará por la ubicación donde deseamos guardar la clave (por defecto se guardará en `/home/<usuario>/.ssh/id_rsa`) y la frase de paso con la que queremos proteger dicha clave.
3. A continuación, usamos el comando `ssh-copy-id` acompañado del nombre de usuario al que deseamos acceder y la dirección IP de la máquina remota. Esto instala la clave pública y la asocia a dicho usuario en dicha máquina.
4. Para terminar, ya podemos acceder mediante SSH al usuario de la máquina remota sin que nos exija la autenticación correspondiente. Aquí pueden suceder dos cosas:
  - Si estamos accediendo como usuario normal sin permisos de administración, al ejecutar la conexión SSH el sistema nos pedirá una vez la frase de paso para desbloquear la clave pública creada anteriormente (que no es lo mismo que introducir la clave del usuario en la máquina remota), y ya accederemos a la máquina deseada.
  - Si estamos accediendo como usuario `root`, no nos pedirá dicha frase de paso y tendremos acceso directo a la máquina.



The image shows two terminal windows from a VMware VirtualBox. The left window is titled 'US14.04\_P1 [Corriendo] - Oracle VM VirtualBox' and shows a user named 'amorente' at 'localhost'. The user runs the command `ssh-keygen -b 4096 -t RSA`. The terminal shows the generation of a public/private RSA key pair, saving the public key to `/home/adri/.ssh/id_rsa.pub` and the private key to `/home/adri/.ssh/id_rsa`. The user then runs `ssh-copy-id amorente@10.0.2.5`, which successfully copies the public key to the remote machine. The right window is titled 'CentOS [Corriendo] - Oracle VM VirtualBox' and shows the same user at 'localhost'. The user runs `ifconfig`, displaying network interface details for `enp0s3` and `lo`. The IP address for `enp0s3` is `10.0.2.5`.

```
(lun nov 21-12:44:20)-(adri@us14:Descargas)$ ssh-keygen -b 4096 -t RSA
Generating public/private RSA key pair.
Enter file in which to save the key (/home/adri/.ssh/id_rsa):
/home/adri/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/adri/.ssh/id_rsa.
Your public key has been saved in /home/adri/.ssh/id_rsa.pub.
The key fingerprint is:
51:75:92:17:8b:27:f7:93:c8:9f:5c:06:d7:61:65:7b adri@us14
The key's randomart image is:
+--[ RSA 4096 ]-----+
|.0.0+|
|. +00+|
|. 0.=E|
|. .+.0|
|S . 0 +|
|. 0.+|
|+|
+-----+
(lun nov 21-12:45:08)-(adri@us14:Descargas)$ ssh-copy-id amorente@10.0.2.5
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to
filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you a
re prompted now it is to install the new keys
amorente@10.0.2.5's password:
Number of key(s) added: 1
Now try logging into the machine, with: "ssh 'amorente@10.0.2.5'"
and check to make sure that only the key(s) you wanted were added.
(lun nov 21-12:45:44)-(adri@us14:Descargas)$ ssh amorente@10.0.2.5
Last login: Mon Nov 21 12:34:48 2016 from 10.0.2.4
[amorente@localhost ~]$ exit
```

```
(lun nov 21-12:44:33)-(amorente@localhost.localdomain:~)$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 10.0.2.5 netmask 255.255.255.0 broadcast 10.0.2.255
inet6 fe80::a00:27ff:fe49:1db9 prefixlen 64 scopeid 0x20<link>
ether 08:00:27:d8:1d:89 txqueuelen 1000 (Ethernet)
RX packets 479 bytes 77290 (75.4 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 486 bytes 72164 (70.4 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 0 (Local Loopback)
RX packets 4 bytes 340 (340.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 4 bytes 340 (340.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
ether 16:72:4a:22:aa:01 txqueuelen 0 (Ethernet)
RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
(lun nov 21-12:45:30)-(amorente@localhost.localdomain:~)$
```

Figura 6.1: Generación de clave privada y acceso de forma remota a través de SSH. - Adrián Morente Gabaldón [21/11/2016]

## 7. ¿Qué archivo es el que contiene la configuración del servicio SSH? ¿Qué parámetro hay que modificar para evitar que el usuario root acceda? Cambie el puerto por defecto y compruebe que puede acceder.

La configuración del servicio SSH se encuentra en el archivo `/etc/ssh/sshd_config`, y obviamente contiene todos los parámetros configurables de dicho servicio. Antes de manipular el fichero de configuración, sería conveniente hacer una copia de seguridad de dicho archivo que pudiéramos restaurar si al configurarlo trastocamos algo que no queríamos y que puede comprometer a la estabilidad o el funcionamiento del sistema. Una vez hecho esto, pasamos a la configuración.

- Para evitar que pueda acceder el usuario root, buscaremos dentro de dicho archivo el apartado de *Autenticación* (*Authentication*, como lo veremos en inglés); y encontramos que el segundo parámetro dentro de éste es *PermitRootLogin* con el atributo puesto a *no*. Lo que tenemos que hacer es simplemente descomentar esta línea para que pase a ser efectiva.

```
#PermitRootLogin no
```

- Para cambiar el puerto por defecto nos iremos a modificar la primera línea de configuración del archivo, justo debajo de los comentarios de introducción. Encontramos la siguiente línea, que solo tenemos que descomentar y cambiar el número de puerto al que nosotros queramos (pero con un valor mayor a 1024, ya que por debajo de este número, todos los puertos están restringidos):

```
#Port 22
```

Cabe destacar que siempre que hagamos alguna de estas modificaciones, tendremos que reiniciar el servicio como vimos en ejercicios anteriores para que dicha modificación entre en vigor. Para este caso práctico, he cambiado el puerto de acceso a SSH en CentOS por el 2222, he añadido dicho puerto al firewall y he reiniciado el servicio (con `systemctl restart sshd.service` en modo root). Además, según vemos en el inicio del archivo `sshd_config`, en CentOS (y sistemas SELinux) necesitaremos ejecutar el siguiente comando para que el servicio acepte otro puerto:

```
semanage port -a -t ssh_port_t -p tcp 2222
```

Dicho comando hará que el servicio SSH acepte solicitudes con el protocolo TCP a través del puerto 2222. En la prueba práctica, aparece un error al ejecutarlo porque ya lo definí antes, pero tuve que volver a hacerlo para poder tomar la captura. Acto seguido vemos que funciona:

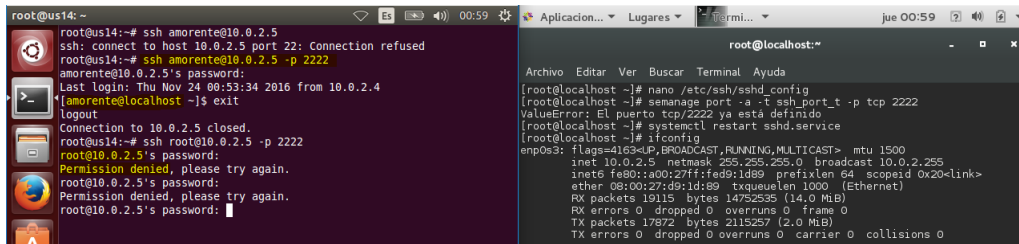


Figura 7.1: Acceso correcto con usuario normal, pero acceso de forma errónea como usuario root. - Adrián Morente Gabaldón [23/11/2016]

## 8. Indique si es necesario reiniciar el servicio ¿Cómo se reinicia un servicio en Ubuntu? ¿y en CentOS? Muestre la secuencia de comandos para hacerlo.

Sí, es necesario reiniciar los servicios al hacer una modificación de este tipo, tanto en Ubuntu Server como en CentOS. En ambos sistemas operativos habremos de tener permisos de administración para ello. Realizamos tal tarea en cada uno de los sistemas operativos con los siguientes comandos:

- **Reiniciar un servicio en Ubuntu:** es fácil de hacer utilizando simplemente el comando `sudo service restart <nombre-servicio>`. Acto seguido, la terminal nos imprime un mensaje confirmando si se ha realizado la acción deseada o no. En la imagen adjunta también utilizo otras utilidades como `stop` y `start` para mostrar cómo detener e iniciar los servicios; (en este caso, SSH y MySQL): - Para un caso práctico más ilustrativo, reinicio los servicios `mysql` y `SSH`, viendo en todo momento el estado actual gracias al mensaje devuelto por la terminal. Acto seguido, a modo de prueba, detengo `SSH` manualmente con `stop` y lo vuelvo a iniciar con `start`.

```
adri@us14:~$ sudo service ssh restart
ssh stop/waiting
ssh start/running, process 11317
adri@us14:~$ sudo service mysql restart
mysql stop/waiting
mysql start/running, process 11346
adri@us14:~$ sudo service ssh stop
ssh stop/waiting
adri@us14:~$ sudo service ssh start
ssh start/running, process 11515
adri@us14:~$
```

Figura 8.1: Reinicio de servicios en Ubuntu Server. - Adrián Morente Gabaldón [20/11/2016]

- Para un caso ilustrativo, detenemos el servicio de HTTP e imprimimos su estado, que como vemos es **dead**. A continuación, mediante la orden vista previamente, reiniciamos el servicio y volvemos a imprimir su estado, siendo **running** esta vez.

- **Reiniciar un servicio en CentOS:** en CentOS también es sencillo manipular servicios, cosa que hacemos mediante el uso de *systemctl* con las diversas opciones que podemos consultar en su propio manual a través de la terminal. En el supuesto práctico, utilizo la opción *stop* para detener el servicio de ejemplo, muestro su estado con *status*, lo reinicio con la opción *restart* y vuelvo a monitorizar su estado.

```
[root@localhost amorente]# systemctl stop httpd.service
[root@localhost amorente]# systemctl status httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor pre
  set: disabled)
   Active: inactive (dead) since dom 2016-11-20 23:21:05 CET; 5s ago
     Docs: man:httpd(8)
           man:apachectl(8)
  Process: 18585 ExecStop=/bin/kill -WINCH ${MAINPID} (code=exited, status=0/
SUCCESS)
  Process: 5533 ExecReload=/usr/sbin/httpd $OPTIONS -k graceful (code=exited,
status=0/SUCCESS)
  Process: 17863 ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND (code=exited
, status=0/SUCCESS)
 Main PID: 17863 (code=exited, status=0/SUCCESS)
   Status: "Total requests: 0; Current requests/sec: 0; Current traffic:  0
B/sec"

nov 20 23:13:05 localhost.localdomain systemd[1]: Starting The Apache HTTP...
nov 20 23:13:05 localhost.localdomain httpd[17863]: AH00558: httpd: Could ...
nov 20 23:13:05 localhost.localdomain systemd[1]: Started The Apache HTTP ...
nov 20 23:21:04 localhost.localdomain systemd[1]: Stopping The Apache HTTP...
nov 20 23:21:05 localhost.localdomain systemd[1]: Stopped The Apache HTTP ...
Hint: Some lines were ellipsized, use -l to show in full.
[root@localhost amorente]# systemctl restart httpd.service
[root@localhost amorente]# systemctl status httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor pre
  set: disabled)
   Active: active (running) since dom 2016-11-20 23:21:34 CET; 1s ago
     Docs: man:httpd(8)
           man:apachectl(8)
  Process: 18585 ExecStop=/bin/kill -WINCH ${MAINPID} (code=exited, status=0/
SUCCESS)
```

Figura 8.2: Reinicio de servicios en CentOS. - Adrián Morente Gabaldón [22/11/2016]

## 9. Muestre los comandos que ha utilizado en Ubuntu Server y en CentOS (aunque en este último puede utilizar la GUI, en tal caso, realice capturas de pantalla). Compruebe que la instalación ha sido correcta.

Tanto en Ubuntu como en CentOS utilizaremos la terminal para la instalación de LAMP, ya que el proceso es sencillo y más liviano de este modo.

- En Ubuntu Server, con una sola línea conseguiremos instalar a la vez *Apache2* para servir HTTP, *MySQL* para la gestión de nuestra base de datos y *PHP* para la administración de la(s) página(s) web que vayamos a ofrecer. Para ello, utilizamos el siguiente comando (con APT):



```
sudo apt-get install lamp-server^
```

- Por otro lado, en CentOS necesitaremos introducir tres líneas en la terminal para que YUM instale todos los paquetes necesarios (similares a los del punto anterior en Ubuntu Server). El primero instalará el servicio *Apache2*, el segundo instalará *PHP* y el último *MariaDB* para la comunicación con nuestra base de datos:

```
sudo yum install httpd
sudo yum install php
sudo yum install mariadb-server
```

Para finalizar, vemos dos capturas de pantalla mostrando que el servidor web ya está instalado en ambos sistemas operativos. Accediendo a la dirección *localhost* desde el navegador de Ubuntu Server llegamos a esta página. Si leemos el mensaje aportado, comprobamos que Apache2 está correctamente instalado y listo para actuar a modo de servidor web. Para ofrecer una página web distinta a la del ejemplo, tendremos que incluir nuestro *index.html* modificado en el directorio */var/www/html* del que Apache2 toma los datos:

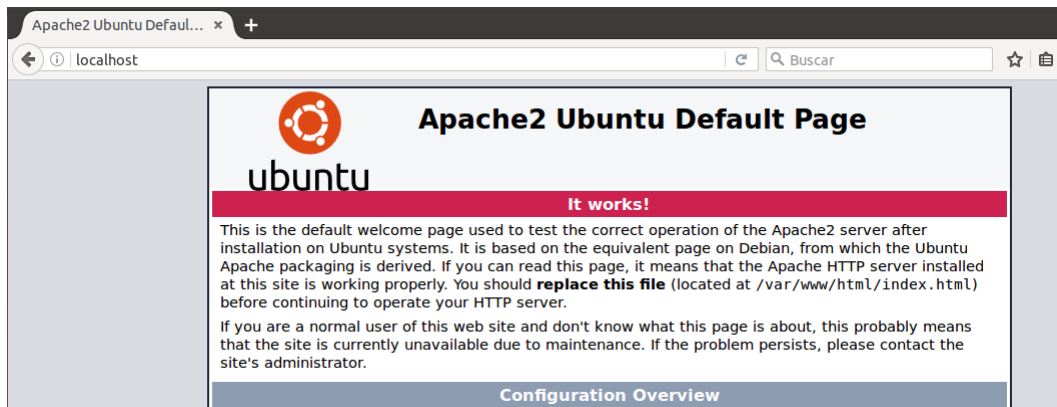


Figura 9.1: Comprobación de la instalación del servicio Apache en Ubuntu Server. - Adrián Morente Gabaldón [22/11/2016]

Accediendo a la dirección *localhost* desde el navegador de CentOS llegamos a esta página. Si leemos el mensaje aportado, comprobamos que Apache está correctamente instalado y listo para actuar a modo de servidor web. Esta es la apariencia que tiene una página web impulsada por Apache y CentOS; por lo que si llegamos a verla se debe a que o no hemos definido ya una página web, o está experimentando problemas (como informa la misma página). Para ofrecer una página web distinta habremos de incluir nuestro *index.html* modificado en el directorio */var/www/html* del que Apache toma los datos:

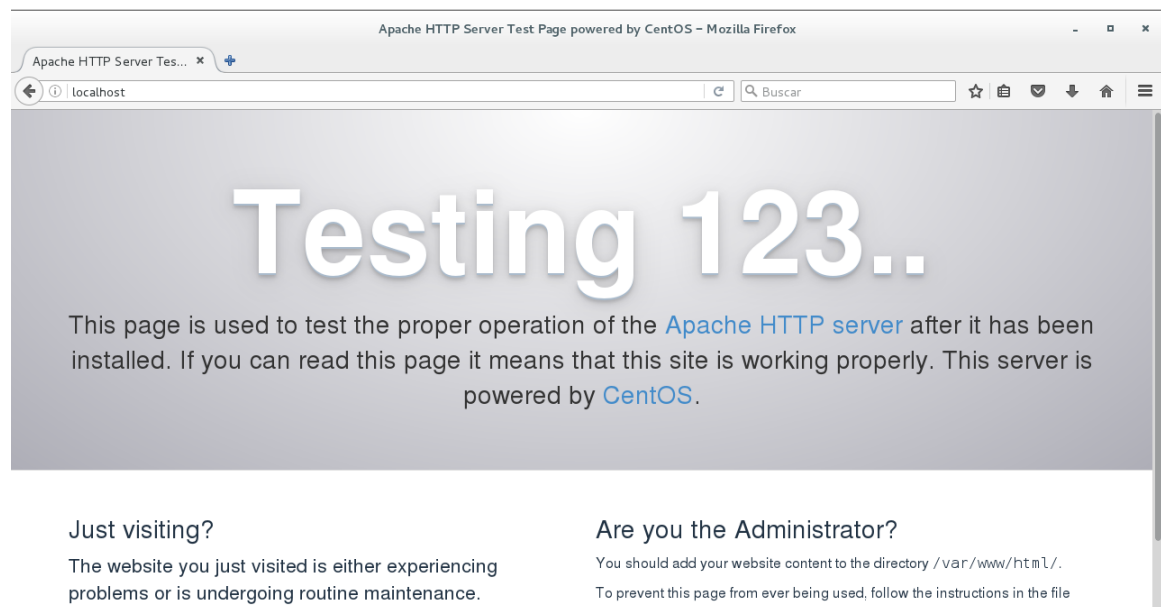


Figura 9.2: Comprobación de la instalación del servicio Apache en CentOS. - Adrián Morente Gabaldón [22/11/2016]

## 10. Realice la instalación usando GUI o PowerShell y compruebe que el servicio está funcionando accediendo a la MV a través de la anfitriona.

Como cualquier instalación de un programa o servicio en Windows, instalar IIS es extremadamente sencillo; tan solo hay que entrar a la configuración inicial del servidor Windows y elegir la función *Agregar roles*. De entre las opciones sugeridas, elegiremos la de IIS junto con todas las características pedidas en el guión de esta práctica.

Una vez que hemos añadido todo lo deseado, apagaremos la máquina, de forma que en VirtualBox podamos añadir un segundo adaptador de red con conexión *Host-Only*. Esto lo hacemos para poder acceder a las direcciones IP de la máquina virtual desde nuestra máquina anfitriona (que en mi caso está corriendo con Windows 10, para un uso más sencillo y estable de la virtualización).

Para comprobar de forma veraz que IIS está funcionando en Windows Server, comenzamos por obtener su dirección IP a través de PowerShell con el comando *ipconfig*. A continuación, introducimos dicha dirección IP en el navegador de la máquina anfitriona; el cual nos mostrará la siguiente página de demostración de IIS:

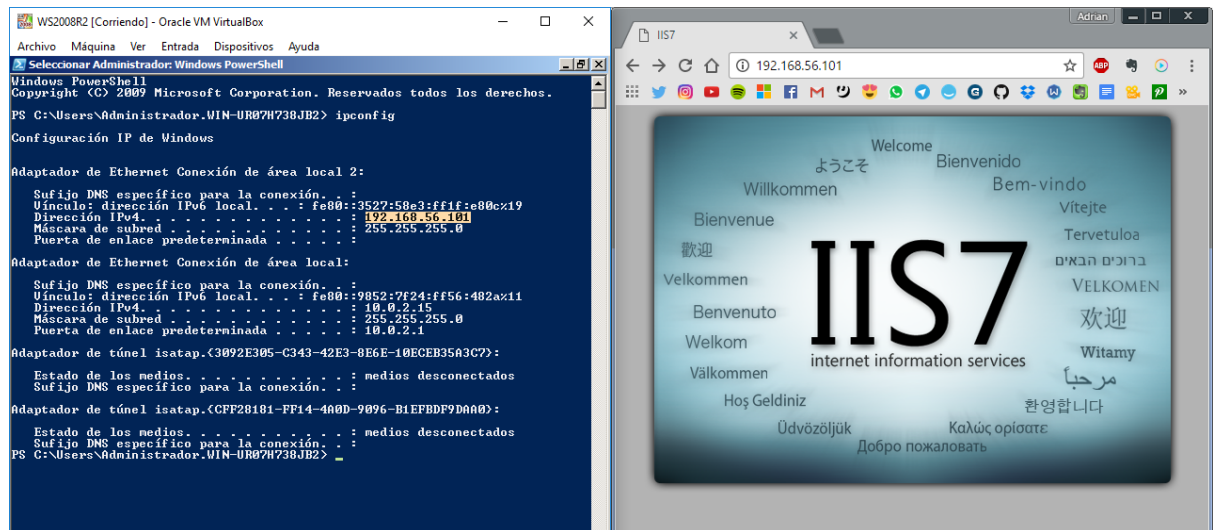


Figura 10.1: Página de demostración del servicio IIS. - Adrián Morente Gabaldón [22/11/2016]

## 11. Muestre un ejemplo de uso del comando patch (p.ej. un programas sencillo HolaMundo con algún fallo concreto [bucle infinito, por ejemplo], y otro programa con dicho fallo arreglado)

El comando patch se utiliza para corregir fallos o aplicar pequeñas modificaciones a un código de alto nivel a partir de un archivo *diff* que incluye dichas modificaciones a realizar. El procedimiento es sencillo, y se basa en:

- Comparar con *diff* dos archivos (uno de ellos el que deseamos corregir, y otro con los errores corregidos), redirigiendo el resultado de la operación a un archivo con extensión *.patch*. Este archivo contendrá las diferencias encontradas entre los dos ficheros.
- Ejecutar el comando *patch* con dos argumentos: uno el archivo pendiente de modificar, y por último el archivo con extensión *.patch* creado en el paso anterior. Esto aplicará el parche al fichero.
- Para terminar, si estamos trabajando con un lenguaje de programación compilado como C o C++, habremos de recompilar el código arreglado. Si se trata de un lenguaje interpretado como Ruby o Python podremos ejecutarlo directamente.

Para este caso práctico, he hecho el ejemplo propuesto en clase y mencionado en el enunciado. Tendremos dos archivos de código en C++, y el primero consta del siguiente contenido:

```
#include <iostream>
using namespace std;
```

```

int main(int argc , char *argv []) {
    for(int i=0; i<5; i--)
        cout << "Hola_mundo!" << endl;
    return 0;
}

```

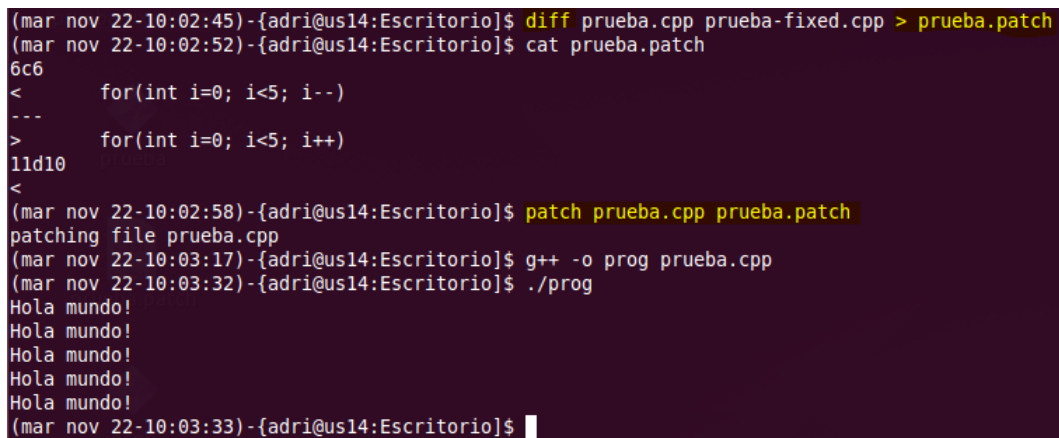
Si nos fijamos un poco, veremos un error en el incremento de la variable interna del bucle for, lo que provoca un bucle infinito. Para solucionar esto, el otro archivo de código tendrá el siguiente contenido:

```

#include <iostream>
using namespace std;
int main(int argc , char *argv []) {
    for(int i=0; i<5; i++)
        cout << "Hola_mundo!" << endl;
    return 0;
}

```

A continuación, una captura de pantalla con el procedimiento mencionado puesto en práctica. En un principio nos encontramos con el fallo en ejecución (no mostrado aquí ya que es un bucle infinito). Comparamos los archivos en busca de las líneas que, al ser diferentes, sean las que potencialmente están causando el problema. Guardamos las diferencias a aplicar en un archivo *patch* y lo aplicamos al archivo erróneo. Para terminar, recompilamos y ejecutamos el nuevo programa para comprobar su funcionamiento.



```

(mar nov 22-10:02:45)-{adri@us14:Escritorio}$ diff prueba.cpp prueba-fixed.cpp > prueba.patch
(mar nov 22-10:02:52)-{adri@us14:Escritorio}$ cat prueba.patch
6c6
<     for(int i=0; i<5; i--)
---
>     for(int i=0; i<5; i++)
11d10
<
(mar nov 22-10:02:58)-{adri@us14:Escritorio}$ patch prueba.cpp prueba.patch
patching file prueba.cpp
(mar nov 22-10:03:17)-{adri@us14:Escritorio}$ g++ -o prog prueba.cpp
(mar nov 22-10:03:32)-{adri@us14:Escritorio}$ ./prog
Hola mundo!
Hola mundo!
Hola mundo!
Hola mundo!
Hola mundo!
(mar nov 22-10:03:33)-{adri@us14:Escritorio}$

```

Figura 11.1: Ejecución de diff-patch con dos programas de prueba. - Adrián Morente Gabaldón [22/11/2016]

## 12. Realice la instalación de esta aplicación (Webmin) y pruebe a modificar algún parámetro de algún servicio. Muestre las capturas de pantalla pertinentes así como el proceso de instalación.

Para la instalación de Webmin en Ubuntu Server nos hemos guiado por el manual oficial de su página web [6]. Hay dos formas de hacerlo en Ubuntu:

- Descargando el archivo completo compilado con extensión *.deb* (el formato admitido por las distribuciones de Linux derivadas de Debian como Ubuntu) e instalándolo ya sea con interfaz gráfica o con el siguiente comando (la X se refiere al número de versión, que podrá variar según cuando descarguemos el archivo y si ha habido actualizaciones):

```
sudo dpkg -i webmin_X_all.deb
```

- La otra forma, por la que nos decantaremos, es utilizar el repositorio APT de Webmin. Un aspecto positivo de este método de instalación es que APT resuelve automáticamente las dependencias necesarias, mientras que *dpkg* o el instalador gráfico podrían no hacerlo. El procedimiento para usarlo es:
  - Para empezar, añadiremos el repositorio aportado por Webmin al archivo */etc/apt/sources.list* que, como sabemos, es el archivo de repositorios en los que busca APT los paquetes a instalar. Antes de hacer una modificación directa del archivo, es conveniente hacer una copia de seguridad del archivo por si tocamos algo que no queremos.
  - A continuación, instalamos la firma GPG del distribuidor de Webmin. Esta es una forma de aportar seguridad a los usuarios que instalan paquetes, ya que reciben un paquete firmado con un nombre propio, que a priori debe ser más seguro que cualquier programa anónimo que encontremos en otro sitio.
  - Recargamos la caché de repositorios de APT e instalamos el paquete *webmin*.

La ejecución práctica de estos pasos previamente detallados es esta:

```
cp /etc/apt/sources.list /etc/apt/sources.list.bak
# Webmin APT Repository >> /etc/apt/sources.list
deb http://download.webmin.com/download/repository sarge contrib
>> /etc/apt/sources.list
cd /root
wget http://www.webmin.com/jcameron-key.asc
apt-key add jcameron-key.asc
apt-get update
apt-get install webmin
```

Hecho todo esto, abriremos el puerto 10000 en el que trabaja Webmin y nos iremos al navegador a la dirección `https://localhost:10000`. Iniciaremos sesión y llegaremos a la pantalla de demostración de Webmin, en la cual encontramos una completa descripción del sistema, ya sea de la fecha actual, las prestaciones de nuestra máquina y el espacio disponible entre otras cosas.

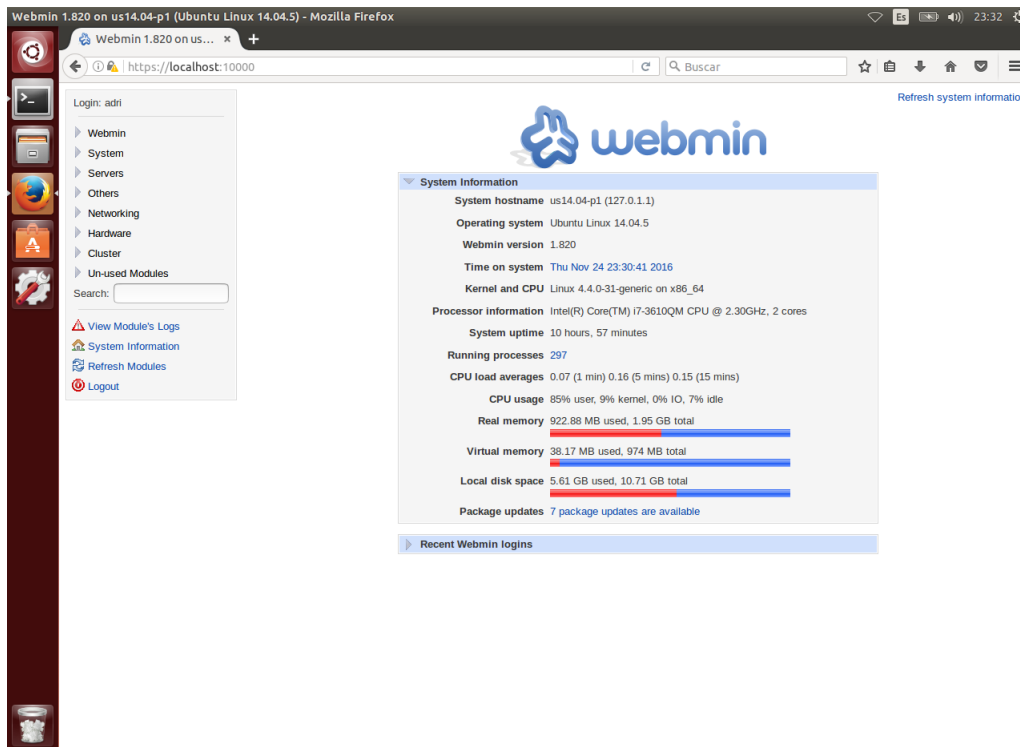


Figura 12.1: Demostración del funcionamiento de Webmin. - Adrián Morente Gabaldón [24/11/2016]

En la columna de la izquierda, Webmin nos propone muchas opciones que podremos configurar. Nos iremos al apartado de Webmin -> Webmin Configuration -> IP Access Control para hacer un ejemplo de configuración. Con la siguiente configuración bloquearíamos el acceso proveniente de direcciones que no figurasen en la lista definida. Es decir, en este caso práctico estaríamos limitando a que solo nuestra máquina de CentOS (con dirección 10.0.2.5 a través de nuestra red NAT) tuviese acceso a nuestra máquina.:

## IP Access Control

The Webmin server can be configured to deny or allow access only from certain IP addresses using this form. Hostnames (like foo.bar.com) and IP networks (like 10.254.3.0 or 10.254.1.0/255.255.255.128 or 10.254.1.0/25 or 10.254.1.5-10.254.97.127) can also be entered. You should limit access to your server to trusted addresses, especially if it is accessible from the Internet. Otherwise, anyone who guesses your password will have complete control of your system.

Figura 12.2: Configuración de bloqueo del acceso al servicio para ciertas direcciones IP. - Adrián Morente Gabaldón [24/11/2016]

### 13. Instale phpMyAdmin, indique cómo lo ha realizado y muestre algunas capturas de pantalla. Configure PHP para poder importar BDs de hasta 25MiB (en vez de los 8 MiB de límite por defecto). Indique cómo ha realizado el proceso y muestre capturas de pantalla.

Como vemos en su página oficial [5], la instalación de phpMyAdmin es fácilmente realizable en Ubuntu (véase la sección de instalación en Debian) con el comando `sudo apt-get install phpmyadmin`. El gestor APT comenzará a descargar y a instalar paquetes, y acto seguido nos mostrará la siguiente imagen:

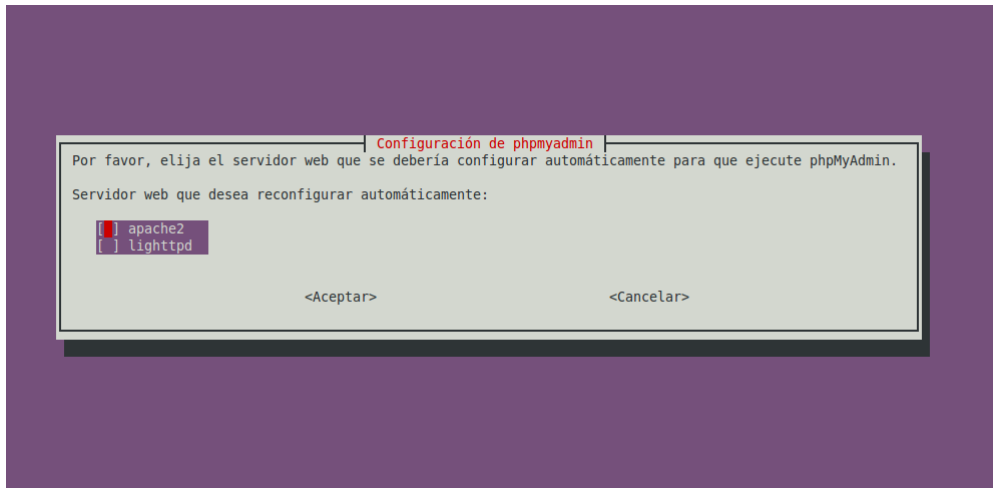


Figura 13.1: Elección de Apache2 para la configuración del servidor web con phpMyAdmin. - Adrián Morente Gabaldón [23/11/2016]

En nuestro caso elegiremos *apache2* (señalándolo y marcándolo con la barra espaciadora), ya que es el servidor web que instalamos antes. A continuación nos pedirá tres contraseñas de protección: una para Apache2, una para la base de datos controlada por MySQL y otra de confirmación. Acto seguido reiniciaremos el servicio Apache2 con el comando que ya conocemos (*service apache2 restart*), e introduciremos la siguiente dirección en nuestro navegador para comprobar que phpMyAdmin está correctamente instalado:

`http://localhost/phpmyadmin`

Al fin estaremos en la pantalla de bienvenida de phpMyAdmin. Introducimos el usuario (*root*) y la contraseña que pusimos durante la instalación, y llegaremos a la pantalla de administración, que nos ofrece todas las posibilidades que vemos:



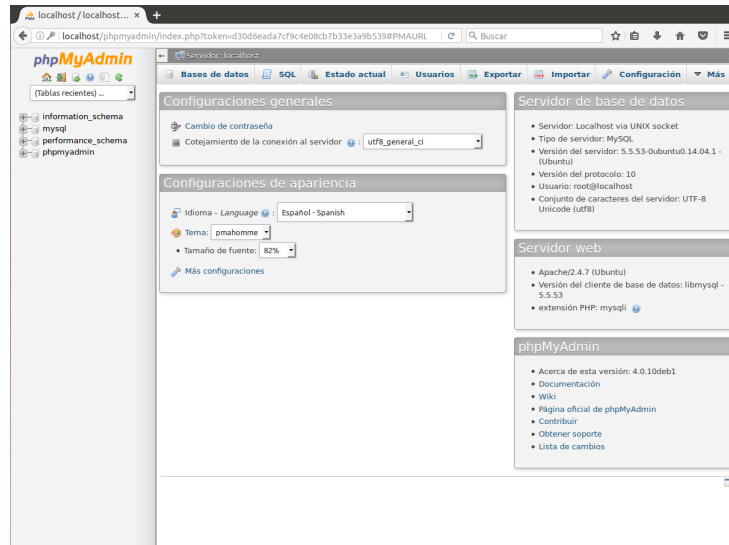


Figura 13.2: Pantalla de inicio del servicio PHPMyAdmin. - Adrián Morente Gabaldón [23/11/2016]

Para terminar, vamos a cambiar el tamaño máximo de subida permitido por el sistema hasta 25MiB. Para ello, accederemos al fichero de configuración `/etc/php5/apache2/php.ini`, que contiene los parámetros modificables de PHP directamente mostrables en el servidor de Apache. Lo podemos abrir con nano en la terminal y buscar `post_max_size`, lo que nos mostrará el siguiente fragmento del archivo. Si leemos el texto descriptivo, nos confirma que esta variable define el tamaño máximo aceptado por PHP; pudiendo fijarse a 0 si quisiéramos no establecer ningún límite. Tras la impresión, lo cambiamos a 25MiB.

```
; Maximum size of POST data that PHP will accept.
; Its value may be 0 to disable the limit. It is ignored if POST data reading
; is disabled through enable_post_data_reading.
; http://php.net/post-max-size
post_max_size = 8M
```

Figura 13.3: Configuración del límite de tamaño de bases de datos en PHPMyAdmin. - Adrián Morente Gabaldón [22/11/2016]

## 14. Visite al menos una de las webs de los software mencionados y pruebe las demos que ofrecen realizando capturas de pantalla y comentando qué está realizando.

Para esta prueba, veremos el administrador ISPConfig. Si entramos en su página web nos encontramos con unas demostraciones que ofrecen bastante completas [4]. Si miramos la barra de menú, nos encontramos con varias pestañas, cada una de las cuales nos permite

gestionar una parte del servidor: podemos añadir clientes, traductores DNS, dominios de correo electrónico, cambiar datos como contraseña y lenguaje, etc.:

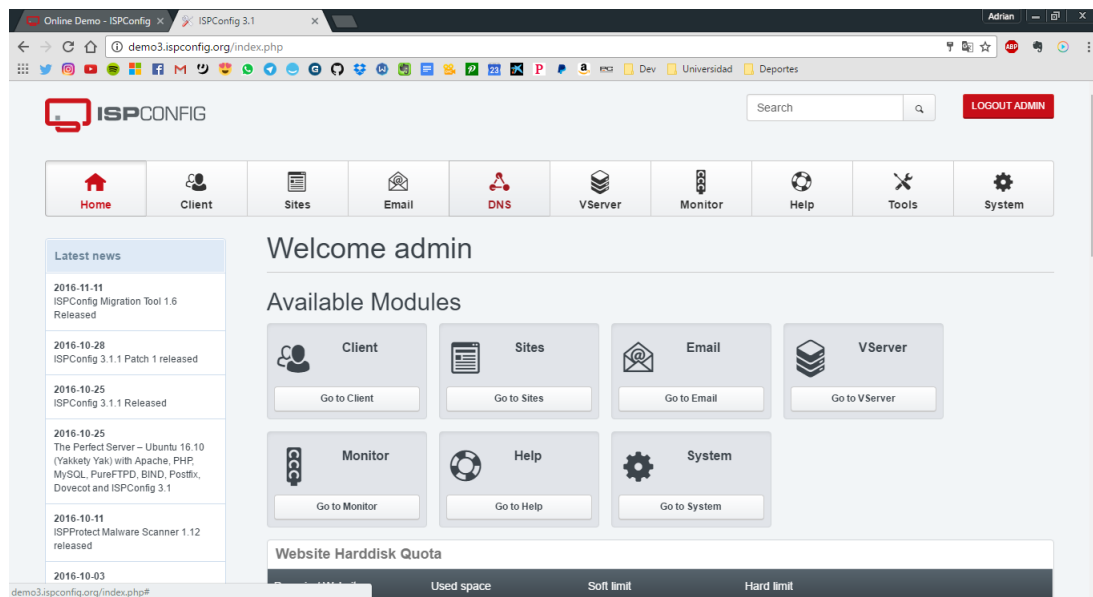


Figura 14.1: Página de inicio de prueba de ISPConfig. - Adrián Morente Gabaldón [22/11/2016]

Para probar una de las funcionalidades ofrecidas, vamos a ver cómo monitorizaríamos uno de los servidores administrados. En la pestaña Monitor encontramos la lista de servidores administrados y el periodo con el que queremos refrescar las estadísticas del profiler usado. En este caso solo tendremos un servidor de prueba ya que, como sabemos, se trata de una demostración gratuita de un servicio de pago. Sin embargo, es ilustrativo en cuanto a las opciones que podemos usar; si miramos el margen izquierdo, encontramos las opciones de monitorización de procesador, CPU, estado de una configuración de discos en RAID, etc. En este caso no podremos cambiar nada por la naturaleza ficticia de la página.

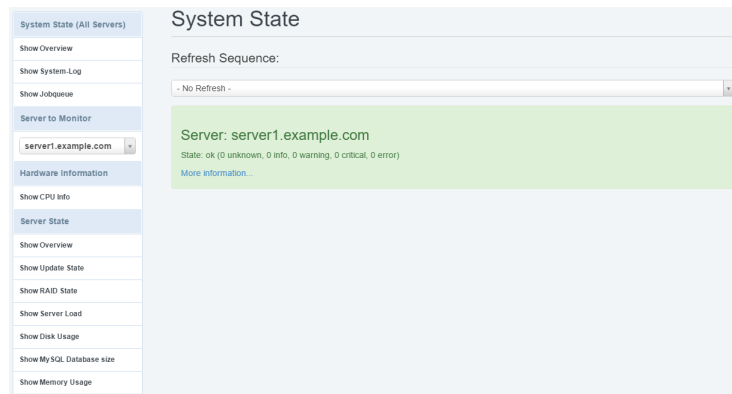


Figura 14.2: Opciones de monitorización de ISPConfig. - Adrián Morente Gabaldón [22/11/2016]

15. a) Ejecute los ejemplos de `find`, `grep` b) Escriba el script que haga uso de `sed` para cambiar la configuración de SSH y reiniciar el servicio. c) Muestre un ejemplo de uso para `awk`.

#### 15.1. Ejemplos de `find` y `grep`

El comando `grep` se utiliza para filtrar cadenas, como bien explica el documento de la práctica. En este caso, hemos ejecutado el ejemplo varias veces, una primera con tres instancias del programa Firefox corriendo, y otra más una vez cerradas. El comando `ps` nos muestra todos los procesos que hay en ejecución en el sistema, que gracias a `grep` filtramos fácilmente. Así nos quedamos con el usuario que está ejecutando el proceso, la hora y el tiempo de ejecución, el directorio y el nombre del proceso, entre otras cosas. Pasemos a ver la captura:

```
[root@localhost ~]# ps -Af | grep firefox
root      26301 15544  0 04:28 pts/0    00:00:00 grep --color=auto firefox
[root@localhost ~]# ps -Af | grep firefox
amorente  26302  3328  33 04:29 ?        00:00:04 /usr/lib64/firefox/firefox
root      26456 15544  0 04:29 pts/0    00:00:00 grep --color=auto firefox
[root@localhost ~]#
```

Figura 15.1: Ejecución de `grep` filtrando el proceso *firefox*. - Adrián Morente Gabaldón [23/11/2016]

Por otro lado tenemos el comando `find`, que lo utilizamos para buscar archivos por el sistema y realizar las acciones que queramos sobre ellos. En cuanto al ejemplo propuesto por la práctica, empezamos listando el contenido de nuestro directorio `/home` y de las carpetas involucradas (`/docs` y `/PDFs`). A continuación, ejecutamos el comando de prueba y vemos como `find` busca los archivos que cumplen el requisito (en este caso, que su nombre termine

en *pdf*) y emprende con ellos la acción determinada: copiarlos a la carpeta /PDFs. Una vez hecho, listamos el contenido de esta carpeta para demostrar su efectividad y sencillez de uso.

```
[amorente@localhost ~]$ ls
Descargas docs Documentos Escritorio Imágenes Música PDFs Plantillas Público Vídeos
[amorente@localhost ~]$ ls docs
1.pdf 2.pdf 3.pdf 4.pdf
[amorente@localhost ~]$ ls PDFs/
[amorente@localhost ~]$ find /home/amorente/docs/ -name '*.pdf' -exec cp {} ~/PDFs \;
[amorente@localhost ~]$ ls PDFs/
1.pdf 2.pdf 3.pdf 4.pdf
[amorente@localhost ~]$
```

Figura 15.2: Ejecución de find moviendo ficheros entre directorios. - Adrián Morente Gabaldón [23/11/2016]

## 15.2. Script de sed para modificar SSH

El comando sed nos ayudará a sustituir caracteres o palabras completas por los argumentos que le introduzcamos. Como siempre que modificamos un archivo importante de configuración, es conveniente realizar una copia de seguridad de dicho archivo a uno igual de la forma *archivo.bak*. Pasemos a ver el código implementado en el script para comentarlo:

```
#!/bin/bash
cd /etc/ssh
puerto\_nuevo = $1
if[ $puerto\_nuevo -gt 1024 ]; then
    sed -i "s/22/$puerto\_nuevo/1" sshd_config
    systemctl restart sshd.service
else
    echo 'El puerto no puede ser menor de 1024'
fi
```

Para empezar, definimos como \$1 el argumento que recibiremos en ejecución, el cual contendrá el número de puerto que pretendemos asignar a la configuración de SSH. Este, como sabemos, deberá ser superior al valor de 1024, ya que los puertos por debajo de este número están reservados. Si el número introducido cumple este requisito, llamamos a la herramienta *sed* con los siguientes parámetros:

- **s**: le pedimos a *sed* que sustituya las dos cadenas especificadas a continuación.
- **22**: el número del puerto que queremos modificar por \$1.
- **\$puerto\_nuevo**: el número introducido como parámetro al programa (\$1).
- **1**: le pedimos que sustituya el primer elemento que encuentra.
- **sshd\_config**: el archivo de entrada al que queremos realizar la modificación.

A continuación, vemos un ejemplo de la ejecución. Para empezar, imprimimos el contenido del archivo de configuración para ver su contenido (puerto 22). A continuación, vemos una ejecución del script errónea, ya que intentamos acceder a un puerto reservado. Justo después, le ponemos un puerto correcto y volvemos a imprimirlo para demostrar su funcionamiento.

```
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
[root@localhost amorente]# bash sed.sh 1023
El puerto no puede ser menor de 1024
[root@localhost amorente]# bash sed.sh 8181
[root@localhost amorente]# cat /etc/ssh/sshd_config | head -n 17
#
$OpenBSD: sshd_config,v 1.93 2014/01/10 05:59:19 djm Exp $

# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
Port 8181
[root@localhost amorente]#
```

Figura 15.3: Configuración de SSH a través de la ejecución de un script de sed. - Adrián Morrente Gabaldón [24/11/2016]

### 15.3. Ejemplo de uso de awk

Awk se trata de un lenguaje de programación de procesamiento de ficheros a muy bajo nivel. Con él podemos programar la manipulación del texto interno de un fichero, como puede ser el de configuración de un servicio, como lo vamos a usar nosotros hoy. Con unas pequeñas líneas de código, podemos conseguir extraer un pequeño fragmento de un archivo, filtrando su contenido. Aunque es de complejidad muy sencilla, nos da una idea de para qué podemos usar esta herramienta. A modo de ejemplo comentaremos el script siguiente:

```
awk '/^Port/ { print $2 }' /etc/ssh/sshd_config
```

Con esta línea estamos diciendo a *awk* que busque la palabra *-Port-* en aquellas líneas en las que no tenga nada precediéndola. Sabemos que en dicho fichero, tras esa palabra aparece el número de puerto para SSH; que será lo que imprimamos en *\$2*, que se traduce por la segunda palabra de esa línea. Al final, cerrando el corchete de *awk*, ponemos el archivo en el que queremos buscar. La ejecución de este mini-programa imprime el puerto 22, ya que como sabemos, es el definido por defecto para el servicio SSH.

```

root@us14:~# nano script_awk.sh
root@us14:~# cat script_awk.sh
awk '/^Port/ { print $2 }' /etc/ssh/sshd_config
root@us14:~# bash script_awk.sh
22
root@us14:~# █

```

Figura 15.4: Búsqueda de parámetro de configuración de SSH con la ejecución de un script de awk. - Adrián Morente Gabaldón [24/11/2016]

## 16. Escriba el script para cambiar el acceso a SSH usando PHP o Python.

Para poder modificar el puerto de acceso al servicio SSH, vamos a utilizar un código en Python que se encargue de tomar un parámetro entero mayor de 1024 (ya que como sabemos, los puertos por debajo de dicho valor están protegidos) y sustituirlo en el archivo de configuración. El script consta del siguiente código:

```

import sys
import fileinput

archivo="/etc/ssh/sshd_config"
puerto_a_configurar="Port_22"
arg=sys.argv[1:]

puerto_nuevo="Port_"+" ".join(arg)

for linea in fileinput.input(archivo, inplace=1):
    if puerto_a_configurar in linea:
        linea=linea.replace(puerto_a_configurar, puerto_nuevo)
        sys.stdout.write(linea)

```

Para empezar, definimos la ruta del archivo a configurar, además del puerto configurado inicialmente. A continuación, unimos a la cadena *-Port* - el número de puerto introducido por parámetro. Acto seguido, con la función *replace* de Python sustituimos fácilmente el contenido mencionado del fichero. Justo después, habremos de reiniciar el servicio para que el cambio realizado surta efecto.

Veamos la ejecución del programa propuesto. Empezamos imprimiendo una parte del fichero de configuración para comprobar que su puerto está intacto (22). Acto seguido, ejecutamos el script descrito y volvemos a imprimir dicha información, verificando que la modificación se hizo correctamente. Para finalizar, reiniciamos el servicio para que se apliquen los cambios.

```
[root@localhost amorente]# cat /etc/ssh/sshd_config | head -n 20 | tail -n 5
#
Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
[root@localhost amorente]# python script.py 8181
[root@localhost amorente]# cat /etc/ssh/sshd_config | head -n 20 | tail -n 5
#
Port 8181
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
[root@localhost amorente]# systemctl restart sshd.service
[root@localhost amorente]#
```

Figura 16.1: Configuración del puerto de acceso a SSH a través de un script en Python. - Adrián Morente Gabaldón [24/11/2016]

## 17. Abra una consola de Powershell y pruebe a parar un programa en ejecución, realice capturas de pantalla y comente lo que muestra.

Aunque la utilización de la consola de Windows para administrar el sistema es mucho más limitada y pesada, también podemos naturalmente comprobar qué procesos se están ejecutando en todo momento en nuestra máquina, y detenerlos si fuera necesario. Para detener un proceso en ejecución podemos hacerlo de dos formas, como bien informa el documento de la práctica:

- **Stop-Process -Name <nombre>**: con este comando detendremos un servicio a partir de su nombre. Esta ha sido la opción que he elegido para la muestra práctica que veremos después.
- **Stop-Process -ID <identificador>**: con este comando detenemos un servicio a partir de su identificador de proceso. Con estos datos ya hemos trabajado en asignaturas como Sistemas Operativos, así que sabemos que cada uno de estos datos identifica unívocamente a un proceso como si de su nombre se tratase. Dicho identificador podemos consultarlo con la orden *Get-Process*, que imprime todos los procesos en curso con sus correspondientes UIDs.

Pasemos a ver la ejecución práctica, la cual es extremadamente sencilla. Empezamos obteniendo la lista de procesos con el comando *Get-Process*, y elegimos (por ejemplo) el proceso VBoxTray, el cual se encarga de mantener el icono de las *VirtualBox Guest Additions* a mano en la barra de tareas de Windows. Con el comando mencionado arriba, fácilmente detenemos dicha aplicación. Volvemos a imprimir los procesos en curso para demostrar su detención.

```

642      0      112      304      3      0.08      1884 System
139      11     2852     6384     51     0.08      1884 taskhost
303      19    47800    49724    248    194.00     864 TrustedInstaller
120      10     2200     5400     51     0.13      644 UBoxService
147      10     2076     6620     71     0.13      1432 UBoxTray
140      26     5560    11232     56     0.02      2892 w3wp
81       9      1492     4440     44     0.36      384 wininit
96       7      1756     5060     27     0.28      416 winlogon
45       6     1008     3436     22     0.00      1348 wlm
94       9      1832     6240     70     0.06      3008 wuauclt

PS C:\Users\Administrador.WIN-UR07H738JB2> Stop-Process -Name UBoxTray
PS C:\Users\Administrador.WIN-UR07H738JB2> Get-Process

Handles  NPM(K)  PM(K)  WS(K)  UM(M)  CPU(s)  Id ProcessName
-----
39        5     1928    4484    43     0.03    2400 conhost
393       11     2140    4252    43     0.58    336 csrss
149       10     1944    5544    42     0.64    376 csrss
67        7     1448    4408    49     0.02    1716 dsm
518       35    15100   28804   147     1.00    1752 explorer
0         0         0        24     0     0.00     0 Idle
591       20     4296   10952    41     0.81    480 lsass
144       7     2356    4056    18     0.03    488 lsm
101       11     2496    6764    38     0.23    2600 mscomsvu
94       10     3072    7740    42     0.16    2652 mscomsvu
146       17     3360    7704    60     0.09    784 msdtc
307       31    60200   60488   557     3.53    2516 powershell
217       14     4776    8304    35     0.75    472 services
30        1      420    1072     4     0.14    264 smss
274       19     6356   10900    76     0.06    1084 spoolsv
152       8     6460   12356    37     1.30    1604 sppsvu
313       33     9988   12368    52     0.36    548 svchost
356       14     3872    9164    42     0.67    580 svchost
252       16     3760    7700    34     0.36    700 svchost
312       15     9224   12100    44     0.91    800 svchost
1513      326   191524  127156  427    85.53    836 svchost
247       18     5600   10192    41     0.36    900 svchost
216       16     4300   10456    63     0.27    960 svchost
479       44   19120   18956   104     1.03   1000 svchost
97        10     4476    9068    37     0.08   1112 svchost
133       13     4372    8812    40     0.05   1136 svchost
47        4      932    2812    13     0.00   1180 svchost
145       13     6776   10604    42     0.03   1320 svchost
69        6     1520    4560    25     0.03   1872 svchost
630       0      112     304     3     0.00     4 System
139      11     2852     6384     51     0.08    1884 taskhost
309      19    62600   55448   265    212.22     864 TrustedInstaller
120      10     2200     5400     51     0.13     644 UBoxService
140      26     5540   11212     55     0.02    2892 w3wp
81       9      1492     4440     44     0.36     384 wininit
96       7      1756     5060     27     0.28     416 winlogon
45       6     1008     3436     22     0.00     1348 wlm
94       9      1832     6240     70     0.06    3008 wuauclt

PS C:\Users\Administrador.WIN-UR07H738JB2>

```

Figura 17.1: Detención de un proceso en Windows Server 2008R2 mediante PowerShell. - Adrián Morente Gabaldón [23/11/2016]

## Referencias

- [1] Comandos de adición de repositorios a yum en centos - página oficial de red-hat. [https://access.redhat.com/documentation/en-US/Red\\_Hat\\_Enterprise\\_Linux/6/html/Deployment\\_Guide/sec-Managing\\_Yum\\_Repositories.html](https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Deployment_Guide/sec-Managing_Yum_Repositories.html). Consultado en 13/11/2016.
- [2] Comandos de configuración de proxy con yum en centos - página oficial de centos. <https://www.centos.org/docs/5/html/yum/sn-yum-proxy-server.html>. Consultado en 13/11/2016.
- [3] Comandos de instalación y eliminación de paquetes de yum - página oficial de centos. [https://www.centos.org/docs/5/html/5.1/Deployment\\_Guide/s1-yum-useful-commands.html](https://www.centos.org/docs/5/html/5.1/Deployment_Guide/s1-yum-useful-commands.html). Consultado en 13/11/2016.
- [4] Demostración del funcionamiento del administrador web ispconfig - página oficial de ispconfig. <http://www.ispconfig.org/ispconfig/online-demo/>. Consultado en 22/11/2016.



- [5] Guía de instalación de phpmyadmin en linux - página oficial de phpmyadmin. <https://docs.phpmyadmin.net/en/latest/setup.html#linux-distributions>. Consultado en 23/11/2016.
- [6] Instalación de webmin en ubuntu server - página oficial de webmin. <http://www.webmin.com/deb.html>. Consultado en 22/11/2016.
- [7] Opciones de uso de secure shell - manual oficial de ssh (openbsd). <https://www.openssh.com/manual.html>. Consultado en 20/11/2016.