

Sesión 3

Habilitando y deshabilitando el login de la cuenta del “root”:

Recordemos de la guía “1.Preparación del Entorno de Laboratorio Linux” (**Pag.43**):

Pregunta: Profesor, y no se puede entrar directamente al usuario “root”, así como hicimos con el usuario “jeanleonardo”, es decir directamente? Es necesario todavía tener que entrar primero al usuario “jeanleonardo” y colocar sudo su?

Respuesta: La contraseña de usuario raíz está deshabilitada, pero eso no significa que se haya eliminado su cuenta raíz de Ubuntu. **No es posible iniciar sesión como usuario raíz porque no se han establecido contraseñas para la cuenta raíz.** Para habilitar la contraseña raíz de Ubuntu, que **está deshabilitada de forma predeterminada por razones de seguridad, debe configurar la contraseña raíz de Ubuntu.** Al ingresar la contraseña, asegúrese de usar diferentes números, caracteres especiales o letras para crear una contraseña segura. Por lo tanto, configurar una combinación única de contraseñas raíz es crucial para mantener la seguridad de su cuenta.

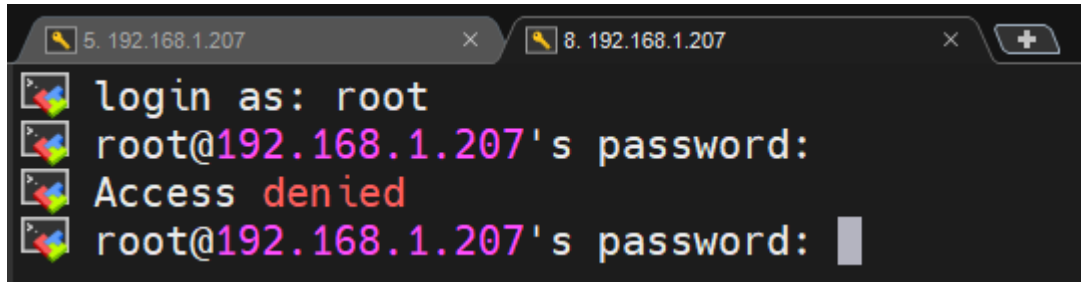
Recordar también: **sudo** sirve para elevar privilegios temporalmente, con el fin de ejecutar comandos o archivos que requieren permisos especiales

La habilitación del login para la cuenta del usuario “root” la realizamos estableciendo una contraseña a esta cuenta, esto lo hacemos de la siguiente forma:

Comando o Instrucción	Explicación o comentario
Nos logueamos con el usuario común “jeanleonardo”	
<p>\$ sudo passwd root</p> <p>Primero, se le pedirá la contraseña del usuario “jeanleonardo” para confirmar la acción con privilegios elevados.</p> <p>Segundo, por ejemplo, asigne la contraseña: otiuni</p>	<p>Recordar que usábamos passwd para cambiar o asignar una contraseña a una cuenta de usuario.</p> <p>Nota: En un entorno real o servidor en producción, esta contraseña tiene que ser mejor elaborada, para evitar problemas de seguridad o ataques de fuerza bruta, estos suceden cuando el atacante emplea determinadas técnicas para probar combinaciones de contraseñas con el objetivo de descubrir las credenciales de una potencial víctima y así lograr acceso a una cuenta o sistema.</p>
\$ logout	Nos “deslogueamos” y probemos entrar “directamente” con el usuario root

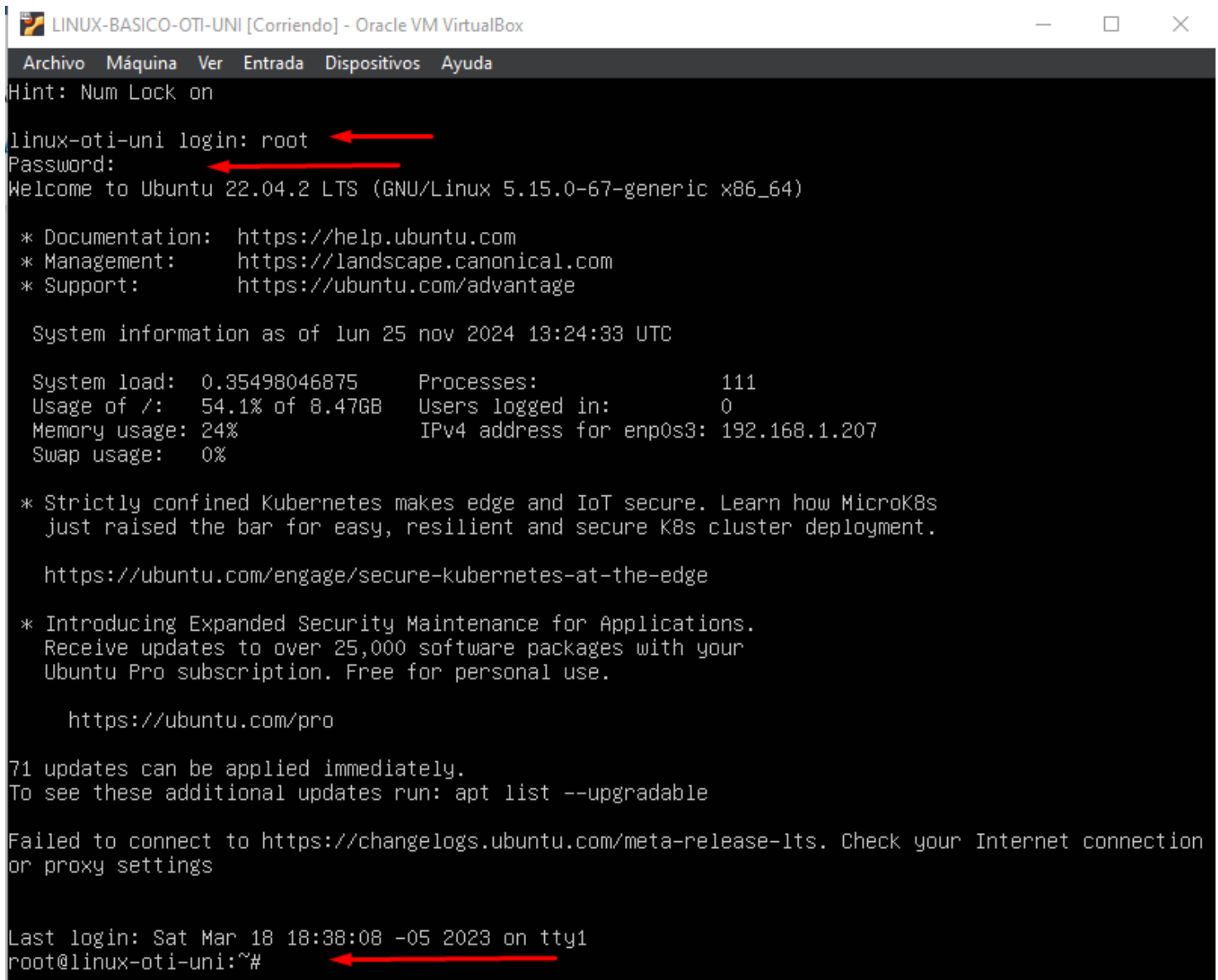
Al usted querer loguearse por SSH con el usuario root y la contraseña asignada, le saldrá “Access Denied”, mientras que, si lo realiza directamente desde la máquina virtual, verá que podrá ingresar sin problemas al usuario root, veamos:

Desde MobaXterm (Via SSH) :



```
login as: root
root@192.168.1.207's password:
Access denied
root@192.168.1.207's password: 
```

Desde la misma máquina virtual (No SSH, conexión directa):



```
LINUX-BASICO-OTI-UNI [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
Hint: Num Lock on

linux-oti-uni login: root
Password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.15.0-67-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

System information as of lun 25 nov 2024 13:24:33 UTC

System load:  0.35498046875   Processes:            111
Usage of /:   54.1% of 8.47GB   Users logged in:      0
Memory usage: 24%             IPv4 address for enp0s3: 192.168.1.207
Swap usage:   0%

* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
  just raised the bar for easy, resilient and secure K8s cluster deployment.

  https://ubuntu.com/engage/secure-kubernetes-at-the-edge

* Introducing Expanded Security Maintenance for Applications.
  Receive updates to over 25,000 software packages with your
  Ubuntu Pro subscription. Free for personal use.

  https://ubuntu.com/pro

71 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection
or proxy settings

Last login: Sat Mar 18 18:38:08 -05 2023 on tty1
root@linux-oti-uni:~#
```

Nota: Para desloguearse del usuario root, se utiliza “exit” no “logout”, este último sirve para desloguearse de usuarios comunes.

Este problema ocurre debido a que de forma predeterminada, no puede iniciar sesión en la cuenta raíz a través de SSH en Ubuntu 22.04. Esta es una función de seguridad, porque no querrá que alguien obtenga acceso de root a su servidor a través de la **fuerza bruta** de la contraseña de root en SSH. Sin embargo, es bastante fácil habilitar el inicio de sesión raíz si desea renunciar a esta recomendación de seguridad.

Permitir el inicio de sesión raíz SSH en Ubuntu 22.04 instrucciones paso a paso:

1. Comience abriendo una terminal de línea de comandos y abriendo el archivo de configuración SSH → /etc/ssh/sshd_config con vi, nano o su editor de texto preferido. Asegúrese de hacer esto con permisos de root.

Comando o Instrucción	Explicación o comentario
Nos logueamos con el usuario común “jeanleonardo”	
\$ sudo nano /etc/ssh/sshd_config	

2. Dentro de este archivo, debemos descomentar **#PermitRootLogin prohibit-password** y cambiar la configuración a **Yes**. Debería obtener lo siguiente:

Antes:

```
# Authentication:

#LoginGraceTime 2m
#PermitRootLogin prohibit-password ←
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

Después:

```
# Authentication:

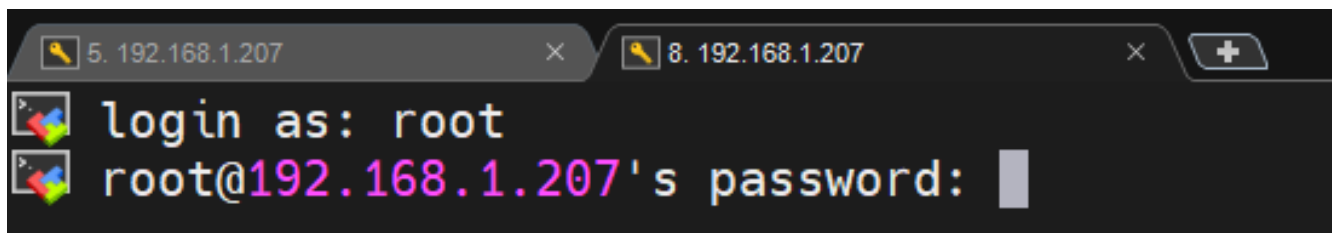
#LoginGraceTime 2m
PermitRootLogin yes ←
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

Guarde los cambios con “:wq!” en **vi** o “Ctrl + Z” y luego Enter en **nano**

3. Ahora debemos reiniciar el servicio SSH para que los cambios surtan efecto.

Comando o Instrucción	Explicación o comentario
\$ sudo systemctl restart ssh	

Después de esto, ya podremos ser capaces de loguearnos “directamente” con el usuario “root” a través de SSH utilizando MobaXterm:



Efectivamente pudimos loguearnos con el usuario “root”:

```

5. 192.168.1.207 x 8. 192.168.1.207 x +
Memory usage: 24% IPv4 address for enp0s3: 192.168.1.207
Swap usage: 0%

* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
just raised the bar for easy, resilient and secure K8s cluster deployment.

https://ubuntu.com/engage/secure-kubernetes-at-the-edge

* Introducing Expanded Security Maintenance for Applications.
Receive updates to over 25,000 software packages with your
Ubuntu Pro subscription. Free for personal use.

https://ubuntu.com/pro

71 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet co
nnection or proxy settings

Last login: Sat Mar 18 18:41:11 2023
/usr/bin/xauth: file /root/.Xauthority does not exist
root@linux-oti-uni:~#
  
```

Nota: Si desea deshabilitar el login por SSH para el usuario root, simplemente siga el proceso inverso al detallado en esta guía.

La deshabilitación del login para la cuenta del usuario “root” de la siguiente forma:

Comando o Instrucción	Explicación o comentario
Nos logueamos con el usuario común “jeanleonardo”	
\$ sudo passwd -l root Le pedirá la contraseña del usuario “jeanleonardo” para confirmar la acción con privilegios elevados.	Deshabilitación del login para la cuenta del usuario “root”

Compruebe, tratando de ingresar desde MobaXterm y la máquina virtual con el usuario root, verá que ya no podrá hacerlo. **Luego de esto, habilite el “login” para la cuenta del usuario “root”, para propósitos del curso vamos a trabajar de esta manera.**

Agregando usuarios administrativos o privilegios administrativos a un usuario común:

Recordar cuando instalamos Ubuntu Server, en el proceso de instalación creamos un usuario, que en mi caso fue “jeanleonardo”, por defecto este será el único usuario que tendrá privilegios administrativos, esto es porque pertenece al grupo “sudo”, para el usuario “jeanleonardo” el grupo sudo es un grupo secundario. Veamos:

```
jeanleonardo@linux-oti-uni:~$ id jeanleonardo
uid=1000(jeanleonardo) gid=1000(jeanleonardo) groups=1000(jeanleonardo),4(adm),24(cdrom),27(sudo),30(dip),46(lxd)
jeanleonardo@linux-oti-uni:~$
```

Vemos que el usuario común “jeanleonardo” pertenece al grupo “sudo”.

Los usuarios que pertenecen al grupo **sudo** tienen acceso a “**sudo**”.

“**sudo**” es un programa diseñado para permitir que el administrador del sistema (usuario root) permita que algunos usuarios ejecuten algunos (o todos los) comandos como el usuario root (u otro usuario, por defecto es el usuario root)

Pregunta: Profesor y el grupo “adm”, que se observa en la imagen anterior, tiene que ver algo con privilegios de administración?

Respuesta: No, este se usara para tareas de monitoreo del sistema. Los miembros de este grupo pueden leer muchos archivos de registro (log files) en /var/log y pueden usar xconsole. Históricamente, /var/log era /usr/adm (y más tarde /var/adm), por lo que el nombre del grupo

Conclusiones:

- Entonces podemos decir que los usuarios que pertenecen al grupo **sudo** son un grupo de usuarios que tienen acceso privilegiado y puede ejecutar comandos como el usuario root.
- Vemos que bastaría añadir a un usuario común al grupo “sudo” para que pueda tener acceso privilegiado y puede ejecutar comandos como el usuario root.**

Digamos que queremos que el usuario “mgaray” se convierta en usuario administrativo, para esto veamos el antes y después:

Antes:

```
mgaray@linux-oti-uni:~$ sudo su
[sudo] password for mgaray:
mgaray is not in the sudoers file. This incident will be reported.
mgaray@linux-oti-uni:~$
```

En Linux, el archivo sudoers es donde se otorga privilegios elevados a los usuarios. De forma predeterminada, solo el usuario raíz tiene estos privilegios. Puede ver su contenido y editarlo solo con el usuario root (**cat /etc/sudoers**), con un usuario común no podrá ver su contenido ni editarlo. Para editar el contenido del archivo sudoers debe ejecutar el comando “visudo” con el usuario root. Esto abrirá el archivo sudoers en el editor de texto predeterminado en terminal, que es **nano** de forma predeterminada, esto se puede cambiar.

OJO: ¡Nunca edite este archivo con un editor de texto normal! ¡Use siempre el comando visudo en su lugar!

Esto debido a que la sintaxis incorrecta en el archivo /etc/sudoers puede dejarlo con un sistema roto donde es imposible obtener privilegios elevados, es importante usar el comando visudo para editar el archivo. El comando visudo abre un editor de texto como de costumbre, pero valida la sintaxis del archivo al guardarlo. Esto evita que los errores de configuración bloqueen las operaciones de sudo, que pueden ser su única forma de obtener privilegios de root.

Después:

Comando o Instrucción	Explicación o comentario
Ingrese como usuario root	
# usermod -a -G sudo mgaray	Agregando el grupo “sudo” como grupo secundario al usuario “mgaray”, recuerde utilizar “-a -G”, si solo usa “-G” sobrescribiría la lista completa de grupos secundarios, eliminando al usuario mgaray de los grupos secundarios en los que estaba y dejándolo solo con el grupo secundario “sudo”

Comprobamos:

```
mgaray@linux-oti-uni:~$ id mgaray
uid=1003(mgaray) gid=1004(abastecimiento) groups=1004(abastecimiento),27(sudo)
```

Comprobamos ejecutando el comando “sudo su” y que paso? No funciona? Nos sale lo siguiente:

```
mgaray@linux-oti-uni:~$ sudo su
[sudo] password for mgaray:
mgaray is not in the sudoers file. This incident will be reported.
```

Para que los cambios se apliquen o actualicen, tiene que desloguearse del usuario mgaray en MobaXterm e ingresar de nuevo veamos:

```
mgaray@linux-oti-uni:~$ logout

Session stopped
- Press <return> to exit tab
- Press R to restart session
- Press S to save terminal output to file
```

Comprobemos de nuevo:

```
mgaray@linux-oti-uni:~$ sudo su
[sudo] password for mgaray:
root@linux-oti-uni:/home/mgaray#
root@linux-oti-uni:/home/mgaray#
root@linux-oti-uni:/home/mgaray#
root@linux-oti-uni:/home/mgaray# who am i
mgaray pts/4 2023-03-18 20:26 (192.168.1.199)
```

Funcionó.

Recordar: Con el comando “sudo su” entramos o nos logueamos con el usuario root pidiéndonos la contraseña del usuario actual con el que estamos logueados, no nos pide la contraseña del usuario root sino la del usuario actual con el que estamos logueados.

Eliminando a un usuario de todos sus grupos secundarios:

Para esto simplemente tenemos que utilizar el siguiente comando (estando como root):

usermod -G "" NOMBREDEUSUARIO

Por ejemplo:

```
root@linux-oti-uni:/home# id pizquierdo
uid=1013(pizquierdo) gid=1006(despacho) groups=1006(despacho),1002(cuentas),1003(sistemas)
```

El usuario pizquierdo tiene como grupos secundarios a “cuentas” y “sistemas”.

Eliminémoslo de todos sus grupos secundarios:

```
root@linux-oti-uni:/home# usermod -G "" pizquierdo
root@linux-oti-uni:/home# id pizquierdo
uid=1013(pizquierdo) gid=1006(despacho) groups=1006(despacho)
```

Introducción al concepto de máscara (umask):

- Creemos un directorio (**maskrootdirectory**) y un archivo (**maskrootfile**) con el usuario root, observe la cadena de permisos:

```
root@linux-oti-uni:~# mkdir maskrootdirectory
root@linux-oti-uni:~# touch maskrootfile
root@linux-oti-uni:~# ls -l
total 40
-rw-r--r-- 1 root root 157 mar 14 21:52 asistencia
drwxr-xr-x 2 root root 4096 mar 14 20:40 compras
drwxr-xr-x 2 root root 4096 mar 14 21:00 Descargas
drwxr-xr-x 2 root root 4096 mar 14 20:49 Escritorio
-rw-r--r-- 1 root root 55 mar 15 16:06 file
-rw-r--r-- 1 root root 31 mar 14 20:46 file3.txt
-rw-r--r-- 1 root root 49 mar 14 21:19 file.bk
drwxr-xr-x 2 root root 4096 mar 19 20:20 maskrootdirectory
-rw-r--r-- 1 root root 0 mar 19 20:20 maskrootfile
```

Si asignamos un valor binario a cada permiso r,w y x tenemos:

r = 1
w = 1
x = 1
- = 0

Observe que un directorio creado por el usuario root por defecto se crea con los permisos:

rwxr-xr-x

pasando a binario tendríamos:

111101101

Tomando por ternas (de 3 en 3) y sabiendo que un numero binario de tres dígitos representa un numero o dígito octal (dígitos del 0 al 7):

755

Conclusión 1:

Un directorio creado por el usuario root por defecto se crea con los permisos **755**

Observe que un archivo creado por el usuario root por defecto se crea con los permisos:

rw-r--r--

pasando a binario tendríamos:

110100100

Tomando por ternas (de 3 en 3) y sabiendo que un numero binario de tres dígitos representa un numero o dígito octal (dígitos del 0 al 7):

644

Conclusión 2:

Un archivo creado por el usuario root por defecto se crea con los permisos **644**

Veamos el valor de la mascara o umask por defecto asignado para el usuario root:

```
root@linux-oti-uni:~# umask
0022
```

El valor de umask es **022** (solo nos interesará los 3 últimos dígitos)

- Ahora situémonos en otro escenario, creemos un directorio (**maskuserdirectory**) y un archivo (**maskuserfile**) con un usuario común, por ejemplo, **jeanleonardo**, observe la cadena de permisos:

```
jeanleonardo@linux-oti-uni:~$ mkdir maskuserdirectory
jeanleonardo@linux-oti-uni:~$ touch maskuserfile
jeanleonardo@linux-oti-uni:~$
jeanleonardo@linux-oti-uni:~$ ls -l
total 4
drwxrwxr-x 2 jeanleonardo jeanleonardo 4096 mar 19 20:32 maskuserdirectory
-rw-rw-r-- 1 jeanleonardo jeanleonardo 0 mar 19 20:33 maskuserfile
```

Observe que un directorio creado por el usuario común, **jeanleonardo**, por defecto se crea con los permisos:

drwxrwxr-x

pasando a binario tendríamos:

111111101

Tomando por ternas (de 3 en 3) y sabiendo que un numero binario de tres dígitos representa un numero o dígito octal (dígitos del 0 al 7):

775

Conclusión 3:

Un directorio creado por un usuario común, por ejemplo, **jeanleonardo**, por defecto se crea con los permisos **775**

Observe que un archivo creado por el usuario común, **jeanleonardo**, por defecto se crea con los permisos:

rw-rw-r--

pasando a binario tendríamos:

110110100

Tomando por ternas (de 3 en 3) y sabiendo que un numero binario de tres dígitos representa un numero o dígito octal (dígitos del 0 al 7):

664

Conclusión 4:

Un archivo creado por un usuario común, por ejemplo, **jeanleonardo**, por defecto se crea con los permisos **664**

Veamos el valor de la máscara o umask por defecto asignado para el usuario común **jeanleonardo**:

```
jeanleonardo@linux-oti-uni:~$ umask
0002
```

El valor de umask es **002** (solo nos interesará los 3 últimos dígitos)

Resumen:

Usuario	Valor de umask	Permisos por defecto asignados al crear un directorio		Permisos por defecto asignados al crear un archivo	
root	022	755	rwxr-xr-x	644	rw-r--r--
jeanleonardo (usuario común)	002	775	rwxrwxr-x	664	rw-rw-r--

UMASK (Máscara del usuario)

Permite establecer permisos a los ficheros por defecto. Esto afecta a la creación de archivos, directorios, enlaces, etc.

Una máscara se compone de 3 dígitos octales, en base a la cual se calculan los permisos finales por defecto que tendrán los archivos y/o directorios creados por un usuario.

Vemos un gráfico que representa mejor este proceso:

En binario *****	----> Numero decimal *****	---> Numero octal *****
000	0	0
001	1	1
010	2	2
011	3	3
100	4	4
101	5	5
110	6	6
111	7	7

- Un numero binario de tres dígitos representa un numero o dígito octal (dígitos del 0 al 7)
- Tomamos la base octal porque únicamente va de 0 a 7.
- El valor de la máscara lo podemos cambiar o modificar, ya veremos después como hacerlo.

Análisis del UMASK:

Nos logueamos con el usuario root.

Veamos el valor de umask (por defecto) que tenemos con el comando "umask":

```
root@linux-oti-uni:~# umask
0022
root@linux-oti-uni:~#
```

Para analizar el valor de la máscara, solo nos interesa las 3 últimas cifras es decir para un umask 0022 solo consideremos "022". Por defecto esta mascara es asignada por default al usuario root.

Pregunta: Profesor y como obtengo permisos finales por defecto que tendran los archivos y/o directorios creados por el usuario ROOT a partir del valor de su mascara?

Respuesta:

Para esto obtener estos permisos, primero debemos saber que:

- Para los directorios, los PERMISOS BASE son 0777 (**777**) (**rw-rwxrwx**)
- Para los archivos, los PERMISOS BASE son 0666 (**666**) (**rw-rw-rw-**) (es decir por defecto los archivos no tienen el permiso de ejecución, si queremos añadirlo tendríamos que hacerlo el comando **chmod ugo+x NOMBREDEARCHIVO**)

Cálculo del permiso final por defecto que tendrán los DIRECTORIOS creados por el usuario ROOT:

```

PERMISOS BASE de DIRECTORIOS = 7   7   7 -
Valor de la mascara o umask   = 0   2   2
                               *****
                               7   5   5
Pasamos a binario:           111 101 101
Asignando los permisos:      rwx r-x r-x

```

Conclusión: Concluimos entonces que los permisos por default para un directorio creado por el usuario root seran "rwx r-x r-x"

Cálculo del permiso final por defecto que tendrán los ARCHIVOS creados por el usuario ROOT:

```

PERMISOS BASE de ARCHIVOS     = 6   6   6 -
Valor de la mascara o umask   = 0   2   2
                               *****
                               6   4   4
Pasamos a binario:           110 100 100
Asignando los permisos:      rw- r-- r--

```

Conclusión: Concluimos entonces que los permisos por default para un archivo creado por el usuario root seran "rw- r-- r--"

Corroboremos lo que nos dice la teoría:

Comando o Instrucción	Explicación o comentario
Ingrese como el usuario root	
# cd /root	Vamos al directorio hogar del usuario root
# mkdir mascara1 # vi mascara2 Hola	Creemos el directorio "mascara1" y el archivo "mascara2"
# ls -l	Verificamos la cadena de permisos de ambos
<pre>drwxr-xr-x 2 root root 4096 mar 19 21:13 mascara1 -rw-r--r-- 1 root root 5 mar 19 21:13 mascara2</pre>	
Aquí podemos corroborar que efectivamente: <ul style="list-style-type: none"> Los permisos por default para un directorio creado por el usuario root son "rwx r-x r-x" Los permisos por default para un archivo creado por el usuario root son "rw- r-- r--" 	

Pregunta: Profesor, y que pasaría si cambio la máscara a 777 con el usuario ROOT o cualquier otro usuario común, como se asignarían los permisos finales?

Respuesta:

Veamos, con umask de 777 tendríamos:

Cálculo del permiso final por defecto que tendrán los DIRECTORIOS creados por el usuario (ROOT o común):

PERMISOS BASE de DIRECTORIOS	=	7	7	7	-
Valor de la mascara o umask	=	7	7	7	

		0	0	0	
Pasamos a binario:		000	000	000	
Asignando los permisos:		---	---	---	

Conclusión: Concluimos entonces que los permisos por default para un directorio creado por el usuario ROOT o común serán "--- --- ---"

Cálculo del permiso final por defecto que tendrán los ARCHIVOS creados por el usuario (ROOT o común):

PERMISOS BASE de ARCHIVOS	=	6	6	6	-
Valor de la mascara o umask	=	7	7	7	

		-1	-1	-1	
Pasamos a binario:		???	???	???	
Asignando los permisos:		???	???	???	

Pregunta: ¿Que paso profesor? Existe permiso "-1"?

Respuesta:

Lo que pasa es que tenemos que ver cómo funciona en realidad la aplicación de permisos en base a **umask**, en realidad no es una "RESTA" lo que se efectúa, **el proceso es el siguiente:**

1. Sea el valor de umask = 777 (para nuestro ejemplo), recordemos que son 3 dígitos octales, entonces:

7 7 7 = 111 111 111 {en binario}

Corroboremos lo que nos dice la teoría:

Comando o Instrucción	Explicación o comentario
# umask 777	Cambiamos la máscara a 777 (podemos estar logueados con un usuario común o el usuario root, en este caso estamos logueados con el usuario root)
# umask	Verificamos el nuevo valor asignado a la máscara
# cd /root	Realicemos pruebas, vamos al directorio /root
# mkdir mascara3 # vi mascara4 Hola	Creemos el directorio "mascara1" y el archivo "mascara2"
# ls -l	Verificamos la cadena de permisos de ambos

```
d----- 2 root root 4096 mar 19 21:34 mascara3
----- 1 root root    5 mar 19 21:35 mascara4
```

Aquí podemos corroborar efectivamente lo que calculamos teóricamente:

Los permisos por default para un directorio creado por el usuario ROOT o común son "-----"

Los permisos por default para un archivo creado por el usuario ROOT o común son "-----"

Nota: Después de la comprobación regrese el valor de umask a 022 para el usuario ROOT.

Cambio o asignación de permisos (comando chmod):

Podemos usar chmod de dos modos: El modo simbólico y el modo octal.

Modo simbólico: (es el que usaremos y vimos en la sesión 2)

chmod [ugoa] [+] [r] [w] [x] ARCHIVO/DIRECTORIO

Carácter para clases de usuarios:

Letra	Significado
u	user, propietario
g	group, grupo
o	other, otros
a	all, todas las clases

Carácter para permisos:

Letra	Significado
r	Permiso de lectura (read); también llamado bit R
w	Permiso de escritura (write); también llamado bit W
x	Permiso de ejecución (execute); también llamado bit X

Para vincular las clases de usuario con sus respectivos permisos se utilizan los siguientes operadores:

Letra	Significado
+	Con el operador "+" se asignan más derechos de archivo a una clase de usuario. Solo se sobrescriben los derechos afectados.
-	El operador "-" retira derechos de acceso a una clase de usuario.
=	Si los permisos de archivo de una clase de usuario se han de renovar, sin importar qué derechos tuvo antes, se usa el operador "=".

Modo octal:

Especifique los permisos de forma octal a través de 3 dígitos, donde cada uno representa una terna binaria.

Opciones del comando chmod:

Código	Opción	Descripción
-R	recursive	El cambio de los derechos de acceso se aplica a todos los archivos y subdirectorios dentro de una carpeta.
-v	verbose	Después del comando se emite un diagnóstico de todos los archivos procesados.
-c	changes	Después del comando se muestra un diagnóstico para todos los archivos que se han modificado.
-f	silent	Se silencian los mensajes de error.

Cambio de propietario (usuario dueño) o grupo propietario de un archivo o directorio:

- Los comandos **chown** y **chgrp** permiten cambiar el **propietario (usuario dueño)** y **grupo propietario** de un archivo o directorio.
- Solo el usuario root puede cambiar el propietario (usuario dueño)
- El grupo propietario puede cambiarse a otro al que pertenezcamos.

Formato o sintaxis:

Comando o Instrucción	Explicación o comentario
# chown [opciones] propietario ARCHIVO/DIRECTORIO	Usamos este comando si únicamente queremos cambiar el propietario o usuario dueño de un archivo o directorio
# chgrp [opciones] grupo ARCHIVO/DIRECTORIO	Usamos este comando si únicamente queremos cambiar el grupo propietario de un archivo o directorio

# chown [opciones] propietario:grupo ARCHIVO/DIRECTORIO	Usamos este comando si queremos cambiar el propietario (usuario dueño) y el grupo propietario de un archivo o directorio
-----------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------

Opciones del comando chown, chgrp:

Código	Opción	Descripción
-R	recursive	El cambio de propietario, grupo propietario o ambos se aplica a todos los archivos y subdirectorios dentro de una carpeta.
-v	verbose	Indica las operaciones que realiza

Notas o consultas teóricas:

Pregunta: Un usuario común puede cambiarse de grupo primario o cambiarse de grupo secundario?

Respuesta: Un usuario común (diferente del root) no puede cambiarse de grupo primario ni de grupo secundario por sí mismo, **el único usuario que lo puede cambiar de grupo primario y secundario es el usuario ROOT.**

Pregunta: Quien puede cambiar los permisos de un archivo y/o directorio (**comando chmod**)?

Respuesta: Esta acción puede ser realiza por **el usuario ROOT y el propietario (usuario dueño) del archivo y/o directorio en cuestión.**

Pregunta: Quien puede cambiar el propietario (usuario dueño) y grupo propietario de un archivo y/o directorio?

Respuesta:

- El usuario root puede cambiar el propietario (usuario dueño) y grupo propietario de un archivo y/o directorio
- El propietario (usuario dueño) de un archivo y/o directorio (o archivo en general de cualquier tipo) solo puede cambiar el grupo propietario de un archivo y/o directorio (o archivo en general de cualquier tipo) a un grupo al cual el propietario (usuario dueño) también pertenece.

Pregunta: Al cambiar de grupo primario a un usuario, afecta el grupo propietario al cual pertenece el directorio hogar del usuario y los directorios o archivos dentro del directorio hogar?

Respuesta: **Pruebe realizando este escenario**, encontrará que si afecta, al cambiar de grupo primario a un usuario, su directorio hogar y todo lo que este dentro, ahora pertenecera o tendrá como grupo primario al NUEVO GRUPO PRIMARIO (al que cambiamos).

Pregunta: Al efectuar el comando "**ls -l**" en el directorio `/home`, los grupos propietarios de cada directorio hogar de los usuarios siempre corresponderán con los grupos primarios de los usuarios en cuestión?

Respuesta:

Si desea saber u obtener el grupo primario de un usuario NO recurra a colocar "**ls -l**" en `/home`, ya que no siempre el grupo propietario del directorio hogar de un usuario será el grupo primario, si desea saber con SEGURIDAD cual es el grupo primario de un usuario, proceda a usar el comando "**id NOMBREDEUSUARIO**"

Pregunta: Cada usuario (root o común) puede cambiar el valor de su umask (máscara) al valor que quiera o hay alguna restricción?

Respuesta: Si, puede cambiarlo al valor que quiera, no hay ninguna restricción.

OJO: No se confunda:

- Cambiar de propietario (usuario dueño) y/o grupo propietario de un archivo y/o directorio se efectúa con los comandos **CHOWN**, **CHGRP**
- Cambiar los permisos de un archivo y/o directorio se efectúa con el comando **CHMOD**
- Cambiar de grupo (primario y/o secundario) a un usuario se efectúa con el comando **USERMOD** seguido de las opciones `-a -G -g` según sea el caso.

En la sesión 2, vimos ejemplo de cómo utilizar **CHMOD**, **USERMOD**, veamos ahora ejemplos de cómo utilizar **CHOWN**, **CHGRP**:

Comando o Instrucción	Explicación o comentario
Ingrese con el usuario fsalgado	
\$ pwd	Nos encontramos en el directorio hogar
\$ mkdir propiedad	
\$ ls -l	Observe que el propietario (usuario dueño) es fsalgado y el grupo propietario es cuentas
Ingrese con el usuario root	
# usermod -a -G despacho fsalgado	Agregamos a fsalgado al grupo secundario despacho
Ingrese con el usuario fsalgado a través de una nueva terminal	Abra una nueva terminal para loguearse con "fsalgado", no utilice la anterior, de no hacerlo no le saldrá lo que haremos con chgrp a continuación.
\$ id fsalgado	Comprobamos
\$ cd	Vamos al directorio hogar del usuario fsalgado
\$ ls -l	Vemos que el directorio propiedad tiene como grupo propietario al grupo cuentas
\$ chgrp despacho propiedad	Cambiamos el grupo propietario del directorio propiedad al grupo despacho,

	que vemos que es un grupo al cual el usuario fsalgado también pertenece
\$ ls -l	Observe que el grupo propietario del directorio propiedad cambió al grupo “despacho”
Ingrese con el usuario root con una nueva terminal o la que ya tenía abierta	
# chown vchachi /home/fsalgado/propiedad	Cambiamos el propietario (usuario dueño) del directorio propiedad , del usuario fsalgado al usuario vchachi
# ls -l /home/fsalgado/	Observe que el propietario (usuario dueño) del propiedad ahora es vchachi
# chown fsalgado:cuentas /home/fsalgado/propiedad	Ahora haremos que el propietario (usuario dueño) es fsalgado y el grupo propietario sea cuentas, es decir dejémoslo como estaba al inicio
# ls -l /home/fsalgado/	Observe que ahora el propietario (usuario dueño) es fsalgado y el grupo propietario es cuentas

Empaquetar y Comprimir archivos (o directorios) en Ubuntu

Diferencia entre “Empaquetar” y “Comprimir”:

Empaquetar: Es agrupar en un solo archivo varios archivos y/o directorios

Comprimir: Es reducir el tamaño de un archivo a través del uso de un algoritmo de compresión.

Nota: Recuerde que cuando se dice “archivo”, se refiere de forma general a cualquier tipo de archivo de los que ya vimos.

Comando o Instrucción	Explicación o comentario
Ingrese con el usuario “jeanleonardo”	
\$ cd	
\$ pwd	
\$ mkdir tar_pruebas_1	
\$ mkdir tar_pruebas_2	
\$ cd tar_pruebas_1	
\$ mkdir directorio_prueba \$ vi archivo1.txt Hola 1 \$ vi archivo2.txt Hola 2 \$ vi archivo3.txt Hola 3 \$ vi test1.txt Hola 4	Creamos dentro de tar_pruebas_1, el directorio “directorio_prueba”, los archivos “archivo1.txt”, “archivo2.txt”, “archivo3.txt”, “test1.txt”, “test2.txt”, “test3.txt”

\$ vi test2.txt Hola 5 \$ vi test3.txt Hola 6	
\$ pwd	
EMPAQUETAR Y DESEMPAQUETAR ARCHIVOS Y/O DIRECTORIOS	
Usaremos tar para empaquetar y desempaquetar archivos. Empaquetar archivos es agrupar varios archivos y/o directorios en un solo archivo.	
\$ tar cvf prueba1.tar archivo1.txt archivo2.txt archivo3.txt Donde: cvf : Parámetros prueba1.tar : Nombre del archivo tar archivo1.txt archivo2.txt archivo3.txt : Nombre de los archivos y/o directorios a empaquetar	Para empaquetar archivos utilizamos los parámetros cvf, que significan lo siguiente: c : crea el archivo empaquetado tar v : muestra una descripción del proceso de empaquetamiento f : indica que lo que va a continuación es el nombre del archivo tar, que en este caso es prueba1.tar
\$ ls -l	Verificamos que los archivos “archivo1.txt.”, “archivo2.txt” y “archivo3.txt” se empaquetaron en el archivo prueba1.tar
\$ mv prueba1.tar /home/jeanleonardo/tar_pruebas_2	Movemos el archivo prueba1.tar dentro del directorio tar_pruebas_2
\$ cd /home/jeanleonardo/tar_pruebas_2	
\$ ls -l	Verificamos que se encuentre prueba1.tar
\$ tar tf prueba1.tar Donde: tf : Parámetros prueba1.tar : Nombre del archivo tar	Vemos el contenido del archivo prueba1.tar Para listar el contenido de un archivo .tar utilizamos los parámetros tf , que significan lo siguiente: t : indica que se va a listar el contenido del archivo .tar f : indica que lo que va a continuación es el nombre del archivo tar, que en este caso es prueba1.tar
\$ tar xvf prueba1.tar Donde: xvf : Parámetros prueba1.tar : Nombre del archivo tar	Ahora veamos como desempaquetar archivos. En este caso desempaquetaremos el archivo prueba1.tar .

	Para desempaquetar archivos utilizamos los parámetros xvf, que significan lo siguiente: x : indica que se va a desempaquetar el archivo tar v : muestra una descripción del proceso de desempaquetamiento f : indica que lo que va a continuación es el nombre del archivo tar, que en este caso es prueba1.tar
\$ ls -l	Verificamos que los archivos “archivo1.txt.”, “archivo2.txt” y “archivo3.txt” se desempaquetaron a partir del archivo prueba1.tar
\$ rm archivo*.*	Eliminamos todos los archivos.
\$ clear; ls -l	Limpiamos la pantalla y listamos el contenido
\$ cd /home/jeanleonardo/tar_pruebas_1 \$ ls -l	Hasta ahora hemos empaquetado puros archivos estándar (tipo “-“), veamos a continuación como empaquetar tanto archivos (tipo “-“) como directorios (archivos tipo “d”).
\$ tar cvf prueba2.tar archivo*.* directorio_prueba Recordar: * → hace referencia a un conjunto de caracteres cualquier ? → hace referencia a un único carácter	Aquí empaquetaremos “archivo1.txt.”, “archivo2.txt” y “archivo3.txt” que caen dentro de la especificación “archivo*.*” y el directorio “ directorio_prueba ”
\$ ls -l	Comprobamos que se creo el archivo prueba2.tar
\$ mv prueba2.tar /home/jeanleonardo/tar_pruebas_2	Movemos el archivo prueba2.tar dentro del directorio tar_pruebas_2 (con el fin de hacer pruebas)
\$ cd /home/jeanleonardo/tar_pruebas_2	
\$ ls -l	Verificamos que se encuentre prueba2.tar
\$ tar tf prueba2.tar	Vemos el contenido del archivo prueba2.tar
\$ tar xvf prueba2.tar	Desempaquetaremos el archivo prueba2.tar
\$ ls -l	Listamos
EMPAQUETAR-COMPRIIR Y DESCOMPRIIR-DEEMPAQUETAR ARCHIVOS Y/O DIRECTORIOS	

<p>Nuevamente usamos tar pero con un procedimiento diferente.</p> <p>Comprimir archivos es reducir el tamaño de archivos y/o directorios por medio de un algoritmo de compresión.</p>	
<pre>\$ cd /home/jeanleonardo/tar_pruebas_1 \$ ls -l</pre>	
<pre>\$ du -h test*.*</pre> <p>En este caso nos aparece que cada archivo ocupa 4Kbytes, es decir los 3 en total ocupan 12Kbytes. Esto es antes de la compresión y el empaquetamiento</p>	Vemos cuanto ocupa en disco los archivos "test1.txt", "test2.txt" y "test3.txt"
<pre>\$ tar cvfz prueba3.tar.gz test*.*</pre> <p>Donde: cvfz : Parámetros prueba3.tar.gz : Nombre del archivo, lo que me indica esto es que primero se empaqueta todo (los 3 archivos test), de ahí el .tar y segundo se comprimió ese único archivo empaquetado con gzip, de ahí el .gz test*.* : Nombre o referencia a los archivos y/o directorios a empaquetar y comprimir.</p>	<p>Para empaquetar y comprimir archivos utilizamos los parámetros cvfz, que significan lo siguiente:</p> <p>c : crea el archivo empaquetado tar</p> <p>v : muestra una descripción del proceso de empaquetamiento</p> <p>f : indica que lo que va a continuación es el nombre del archivo tar, que en este caso es prueba1.tar</p> <p>z : indica que se va a utilizar gzip para comprimir el archivo</p>
<pre># ls -l</pre>	Listamos y comprobamos la creación del archivo prueba3.tar.gz (observe que es un archivo tipo "-", es decir es un archivo estándar o simplemente archivo)
<pre>\$ mv prueba3.tar.gz /home/jeanleonardo/tar_pruebas_2</pre>	
<pre>\$ cd /home/jeanleonardo/tar_pruebas_2 \$ ls -l</pre>	
<pre>\$ du -h prueba3.tar.gz</pre> <p>En este caso nos aparece que este único archivo empaquetado (que tiene los 3 archivos test) y luego comprimido ocupa 4Kbytes</p> <p>Esto es después de la compresión y el empaquetamiento</p>	Verificamos que se encuentre prueba3.tar.gz
<pre>\$ tar tfz prueba3.tar.gz</pre> <p>Donde: tfz : Parámetros prueba3.tar.gz : Nombre del archivo</p>	<p>Vemos o listamos el contenido del archivo prueba3.tar.gz</p> <p>Para listar el contenido del archivo .tar.gz utilizamos los parámetros tfz, que significan lo siguiente:</p> <p>t : indica que se va a listar el contenido del archivo (.tar.gz)</p>

	<p>f : indica que lo que va a continuación es el nombre del archivo (.tar.gz), que en este caso es prueba3.tar.gz</p> <p>z : indica que se va a utilizar gzip para comprimir el archivo</p>
\$ tar xvfz prueba3.tar.gz	<p>Descomprimos y desempaquetaremos el archivo prueba3.tar.gz</p> <p>Para descomprimir y desempaquetar archivos utilizamos los parámetros xvfz, que significan lo siguiente:</p> <p>x : indica que se va a desempaquetar el archivo tar</p> <p>v : muestra una descripción del proceso de desempaquetamiento</p> <p>f : indica que lo que va a continuación es el nombre del archivo tar, que en este caso es prueba3.tar</p> <p>z : indica que se va a utilizar gzip para descomprimir el archivo</p>
\$ ls -l	Listamos y comprobamos que están los 3 archivos "test".
\$ rm -r directorio_prueba test*.*	
\$ cd /home/jeanleonardo/tar_pruebas_1	
\$ ls -l	Hasta ahora hemos empaquetado y comprimido puros archivos estándar (tipo "-"), veamos a continuación como empaquetar y comprimir tanto archivos (tipo "-") como directorios (archivos tipo "d").
\$ tar cvfz prueba4.tar.gz test*.* directorio_prueba	
# ls -l	
\$ mv prueba4.tar.gz /home/jeanleonardo/tar_pruebas_2	
\$ cd /home/jeanleonardo/tar_pruebas_2	
\$ ls -l	
\$ tar tfz prueba4.tar.gz	
\$ tar xvfz prueba4.tar.gz	
\$ ls -l	

Tabla de Comparación		
Parámetros de comparación	GZIP	TAR
Connotación	La herramienta de compresión Gzip, que gestiona la extensión.gz, se utiliza para minimizar el uso de espacio en disco del archivo.	Tar es un archivador de archivos, lo que significa que puede fusionar varios archivos sin comprimirlos en un solo archivo.
Capacidad de compresión	Gzip es un programa que comprime un solo archivo (solo).	Los diferentes archivos se combinan en un solo archivo (tar) con tar.
Basado en la Eficiencia	Zip es un software que archiva y comprime.	Crea un solo archivo a partir de varios archivos; no comprime datos a menos que se use junto con un programa de compresión como gzip.

Gestión y Manejo de Paquetes:

Teoría básica

¿Qué es un paquete?

Es un archivo comprimido y con una estructura establecida que permite ser tratado por herramientas de gestión de software para realizar operaciones como instalar, compilar, eliminar, purgar los archivos de configuración del sistema, actualizar, etc de forma cómoda, segura, estable y centralizada.

Básicamente hay 2 tipos de paquetes:

- **Binarios:** Contienen la información necesaria para reconstruir una aplicación en un sistema nuevo, sin necesidad de encontrarse en la misma computadora.
- **De Código Fuente:** Es un archivo comprimido con el código fuente del paquete. Es necesario compilarlo e instalarlo.

Entre los **paquetes Binarios** podemos mencionar:

DEB: Es el que más nos interesa, dado que tiene que ver con Ubuntu. Contienen ejecutables, archivos de configuración, páginas de información, derechos de copyright y otras documentaciones. Los paquetes Debian se colocan en archivos .deb

RPM: Desarrollado para la distribución de Red Hat, con el fin de crear un sistema fácil de crear e instalar. Actualmente todas las distribuciones basadas en Red Hat ocupan los paquetes RPM

Hay varios paquetes de código fuente, pero el más utilizado o bien conocido tenemos a:

TGZ (TAR GZ): Archivo de paquetes específico para Unix comprimido. Ocupado para contener aplicaciones, y su código fuente, para no tener que crear un tipo de paquete específico para cada

distribución. A diferencia de los paquetes .deb, o .rpm, este no contiene instrucciones particulares de instalación para cada distribución, por lo que la instalación del contenido deberá ser compilado por el usuario.

Algo importante de tener en cuenta en el tema de gestión de paquetes, es sobre los repositorios, para mas información sobre repositorios consulta la guía de la Sesión 2.

Un repositorio es un sitio centralizado donde se almacena y mantiene información digital. En el caso de los repositorios Linux, esta información son programas. Cada distribución tiene unos repositorios (entre ellos el OFICIAL) en los que están almacenados los programas diseñados para esta distribución.

Uno de los sistemas de gestión de paquetes usado en Ubuntu es: APT (es el que utilizaremos para el curso)

APT (Advanced Packaging Tool), fue creado por el proyecto Debian, y por ende, es el que se usa en Ubuntu (dado que es basado en Debian). Se ejecuta en el terminal y su principal novedad es que unifica todas las opciones y parámetros de apt-cache y apt-get. Se incluye en Ubuntu a partir de la 14.04.

Usos de APT:

Instalación de un paquete:

Sintaxis:

```
sudo apt install [nombre_paquete]
```

La ventaja de usar **apt** en lugar del típico **apt-get** (que es otro gestor de paquetes) es que muestra una barra de progreso.

Desinstalación de un paquete:

Sintaxis:

```
sudo apt remove [nombre_paquete]
```

Buscar un paquete:

Sintaxis:

```
sudo apt search [nombre_paquete]
```

Mostrar información detallada de un paquete:

Sintaxis:

```
sudo apt show [nombre_paquete]
```

En el apartado “APT-Manual-Installed:” (solo aparece si el paquete esta instalado) podemos ver si el paquete en cuestión ha sido instalado manualmente con apt (apt install NOMBREDELPAQUETE) a través del mensaje **yes**

Sincronizar lista de paquetes con el repositorio:

Sintaxis:

`sudo apt update`**Actualizar nuevas versiones de paquetes instalados en el sistema:**

Sintaxis:

`sudo apt upgrade`**Editar la listas de repositorios activas**

Sintaxis:

`sudo apt edit-sources`**Liberar espacio de paquetes que quedaron por ahí en el sistema (cache, fuentes, carpetas residuales, etc):**

Sintaxis:

`sudo apt autoremove`

Con autoremove se eliminan aquellos paquetes perdidos, generalmente instalados como dependencias de otras instalaciones, que ya no son necesarios.

Ver paquetes instalados en el sistema:

Sintaxis:

`sudo apt list --installed`**Ver paquetes que pueden actualizarse en el sistema:**

Sintaxis:

`sudo apt list --upgradable`

Por ejemplo, probemos instalando y manejando el paquete **tree**:

Comando o Instrucción	Explicación o comentario
Ingrese con el usuario "jeanleonardo"	
\$ sudo apt update	
\$ sudo apt upgrade	
\$ sudo apt search tree	Buscamos el paquete tree
\$ sudo apt install tree (si ya lo tiene instalado remuévalo con sudo apt remove tree y empiece con los pasos de la guía)	Instalamos tree

\$ sudo apt show tree	Información detallada del paquete Aquí también podemos ver la versión
\$ sudo apt list --installed	Verifique que tree esta instalado en la lista de paquetes instalado
\$ sudo apt remove tree	
\$ sudo apt list --installed	
\$ sudo apt list --installed	Verifique que “tree” ya no este
\$ sudo apt install tree	Deje todo como estaba antes.
\$ sudo apt-cache policy tree	Otra forma de ver la versión instalada del paquete