UD 4
Sistemas operativos. Configuración.

UD4 Evaluación.

Resultado de aprendizaje	Criterios de evaluación	Actividades evaluables	Peso
4. Gestiona sistemas operativos utilizando comandos y herramientas gráficas y evaluando las necesidades del sistema.	a) Se han configurado cuentas de usuario locales y grupos.	Trabajos / Examen	10
	b) Se ha asegurado el acceso al sistema mediante el uso de directivas de cuenta y directivas de contraseñas.	Trabajos / Examen	10
	c) Se ha protegido el acceso a la información mediante el uso de permisos locales.	Trabajos / Examen	10
	d) Se han identificado, arrancado y detenido servicios y procesos.	Trabajos / Examen	10
	e) Se han utilizado comandos para realizar las tareas básicas de configuración del sistema.	Trabajos / Examen	10
	f) Se ha monitorizado el sistema.	Trabajos / Examen	10
	g) Se han instalado y evaluado utilidades para el mantenimiento y optimización del sistema.	Trabajos / Examen	10
	h) Se han evaluado las necesidades del sistema informático en relación con el desarrollo de aplicaciones.	Trabajos / Examen	10

Total 80 puntos, nota mínima para aprobar 40 puntos.

UD4 Contenidos.

– Configuración de usuarios y grupos locales. Usuarios y grupos predeterminados.
– Seguridad de cuentas de usuario.
– Seguridad de contraseñas.
– Acceso a recursos. Permisos locales. Configuración de perfiles locales de usuario.
– Servicios y procesos. Identificación y administración.
– Comandos de sistemas libres y propietarios para realizar tareas básicas de configuración del sistema.
– Herramientas de monitorización del sistema para la evaluación de prestaciones.
– Instalación de utilidades para el mantenimiento y optimización del sistema

Índice

1. Introducción

1. Introducción.

Objetivos:

- Descubrir los fundamentos de gestión de usuarios y gestión de procesos.
- Crear cuentas de usuario locales y grupos.
- Asegurar el acceso al sistema mediante directivas de cuenta y de contraseñas.
- Proteger el acceso a la información mediante permisos locales.
- Conocer diferentes mecanismos para la gestión de procesos.
- Emplear comandos para realizar tareas básicas de configuración y monitorización del sistema.
- Conocer diferentes herramientas para el mantenimiento y optimización del sistema.
- Saber operar con software de automatización de tareas.

1. Introducción.

Glosario:

- Background. Ejecución de procesos en segundo plano.
- Quantum. Espacio de tiempo asignado a cada proceso para ocupar la CPU.
- GID. Número de identificación de grupo único en sistemas Linux.
- Linux-PAM. Sistema centralizado de autenticación de usuarios en Linux.
- Máscara de permisos. Privilegios de los que dispone el propietario, el grupo y el resto de usuarios sobre un objeto del sistema de archivos en sistemas Linux.
- Modo kernel de ejecución. Capacidad de ejecución en modo privilegiado sobre el sistema operativo.
- PCB. Bloque de control de proceso.
- PID. Identificador de proceso.
- Proceso. Instancia de un programa en ejecución.
- Superusuario. Usuario con mayor privilegio sobre un sistema Linux.
- UID. Número de identificación de usuario único en sistemas Linux

1. Introducción.

¿Por qué es importante el dominio de sistemas?

DevOps es un acrónimo inglés de development (desarrollo) y operations (operaciones), que se refiere a una metodología de desarrollo de software que se centra en la comunicación, colaboración e integración entre desarrolladores de software y los profesionales de sistemas en las tecnologías de la información (IT)

Tres ideas clave:

- DevOps es una metodología para creación de software.
- DevOps se basa en la integración entre desarrolladores software y administradores de sistemas.
- DevOps permite fabricar software más rápidamente, con mayor calidad, menor coste y una altísima frecuencia de releases.

1. Introducción.

- https://www.infojobs.net/caceres/devops-junior-caceres/of-i8d69261866469f88a8337ec7b5dff0?applicationOrigin=search-new&page=1&sortBy=RELEVANCE
- https://www.infojobs.net/madrid/consultor-devops-teletrabajo-flexibilidad-horaria/of-i5a1d1c3e624bf7ad0fd2317cc43d11?applicationOrigin=search-new&page=1&sortBy=RELEVANCE
- https://www.infojobs.net/madrid/graduate-program-cloud-devops-engineer/of-i861e22f2614805a760ab4def96c0b5?applicationOrigin=search-new&page=1&sortBy=RELEVANCE

2. Gestión de usuarios por línea de comandos en Linux

2. Gestión de usuarios por línea de comandos en Linux

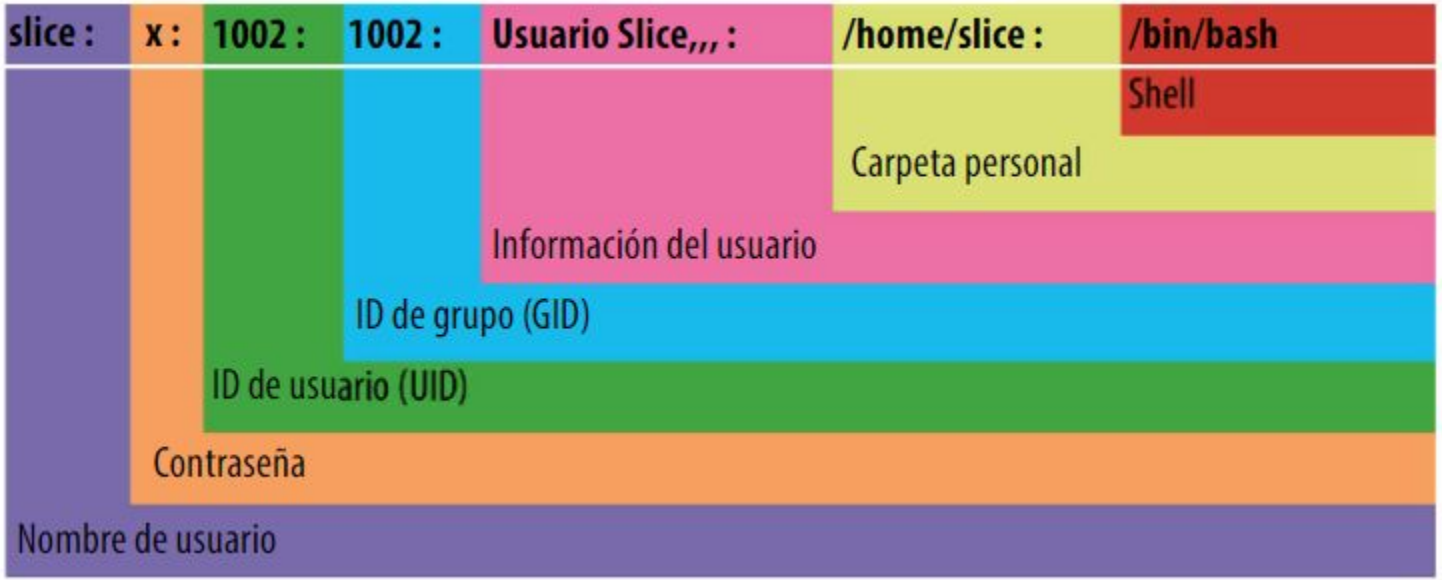
Los sistemas GNU/Linux gestionan los usuarios mediante archivos de configuración.

Los usuarios comunes no gozan de privilegios, por lo que son los administradores o el usuario root los únicos que pueden editarlos.

Los usuarios y grupos en Linux se gestionan a través de los archivos /etc/passwd y /etc/group, principalmente, además de otros muchos, como /etc/sudoers, /etc/shadow, etc.

2. Gestión de usuarios por línea de comandos en Linux

/etc/passwd almacena los usuarios del sistema



2. Gestión de usuarios por línea de comandos en Linux

1. Login de usuario o nombre que se emplea para acceder al sistema.
2. Password requerida para autenticarse en el sistema. Aparece una “x” indicando que la contraseña se encuentra encriptada/hasheada en el fichero de configuración /etc/shadow.
3. UID o número de identificación de usuario único (User IDentification). El 0 se corresponde con el superusuario: del 1 al 99 para cuentas predeterminadas y de la 100 a 999 para cuentas administrativas del sistema, por tanto, los nuevos usuarios serán asociados a partir del 1000.
4. GID o número de identificación del grupo principal del usuario (Group IDentification).
5. Información personal del usuario, donde suelen incluirse datos de localización del usuario, como teléfono, oficina, etc.
6. Home o directorio de trabajo, es decir, el directorio inicial del usuario, cuando se conecta al sistema..
7. Shell o intérprete de comandos empleado por el usuario cuando inicia el sistema. Se puede ver los shells disponibles con cat /etc/shells

2. Gestión de usuarios por línea de comandos en Linux

Por usuario administrador en Linux entendemos aquel que tiene capacidad de gestión en el sistema, sin ser necesariamente el superusuario (root).

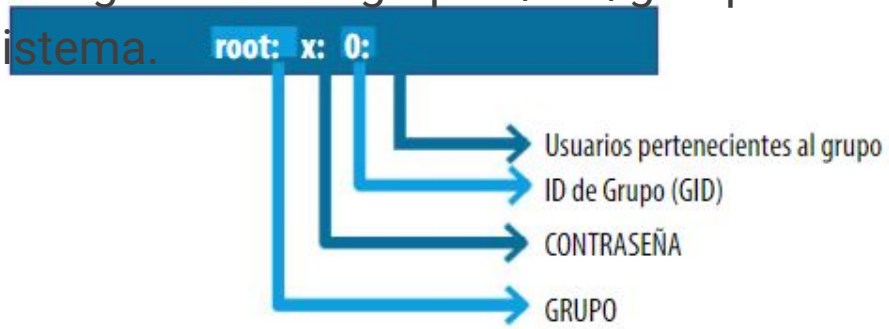
Esta capacidad puede ser desarrollada si dispone de privilegios gracias al comando sudo o si se encuentra en grupos de usuarios con privilegios sobre determinados archivos o comandos de gestión.

2. Gestión de usuarios por línea de comandos en Linux

/etc/group

Los grupos en Linux son muy empleados, ya que facilitan la administración de privilegios en el sistema. Por ejemplo, se emplean cuando se desea que algunos usuarios tengan permisos sobre archivos o carpetas (lectura o edición), sin ser los propietarios de los mismos.

El fichero de configuración de grupos /etc/group centraliza la gestión de grupos en el sistema.



2. Gestión de usuarios por línea de comandos en Linux

1. Nombre del grupo: nombre del grupo asociado al identificador del grupo.
2. Contraseña: no se suele utilizar, apareciendo una “x”, e indica que la contraseña se encuentra encriptada/hasheada en el fichero de configuración /etc/gshadow.
3. Identificador de grupo: GID o número de identificación del grupo único (Group IDentification).
4. Lista de usuarios: usuarios pertenecientes al grupo identificado como grupo secundario. Un usuario puede pertenecer a varios grupos.

Cada usuario ha de pertenecer a un grupo principal (cuarto campo del fichero /etc/passwd), pero, además, puede pertenecer a varios grupos secundarios, especificándose en /etc/group.

2. Gestión de usuarios por línea de comandos en Linux

El **superusuario** dispone de tal control sobre el sistema que, en principio, no está “habilitado”.

El prompt, o la línea de petición de órdenes en el intérprete de comandos, varía según el usuario activo en él. El indicativo de petición en el prompt es el símbolo “#”, mientras que para el resto de usuarios es “$”.

Podemos **cambiar la contraseña** de root con

$ sudo passwd root

Después podrías cambiar al usuario root con

$ su root o sudo su

2. Gestión de usuarios por línea de comandos en Linux

Cambio de usuario

Podemos ejecutar comandos de otro usuario con

$ sudo -u usuario comando_de_usuario

Otra opción es cambiar de usuario con el comando su, visto en la página anterior

$ su - usuario

2. Gestión de usuarios por línea de comandos en Linux

Ejercicio:

1. Indica que usuarios no administrativos tiene tu SO
2. ¿Qué shells aparecen en el fichero passwd?
3. Ejecuta los shell de tss y uuidd
4. ¿Qué grupos secundarios tiene tu usuario?

2. Gestión de usuarios por línea de comandos en Linux

Crear usuarios

useradd [-g grupo] [-G grupo[, grupo …]] [-d directorio_trabajo [-m]] [-p contraseña_encriptada] [-s shell] [-u id_usuario] usuario

Este comando crea una nueva entrada en /etc/passwd y permite copiar los archivos del directorio /etc/skel al directorio de usuario. El directorio /etc/skel contiene los archivos de configuración por defecto que se añaden al directorio de trabajo de un usuario cuando es creado con las opciones adecuadas.

2. Gestión de usuarios por línea de comandos en Linux

Crear usuarios

Opciones:

- g grupo: asignación al grupo principal. Todos los usuarios están adscritos, al menos, a un grupo principal y, en caso de pertenecer a más de uno, el resto serán grupos secundarios. Estos grupos han de existir. En caso de no especificar esta opción, se creará por defecto un grupo con su mismo nombre.
- G grupos: lista de grupos secundarios separados por comas y sin espacios.
- d directorio de trabajo: establece un directorio existente como directorio de trabajo para dicho usuario. Si no se especifica esta opción, se tomará por defecto /home/nombre_usuario.
- p contraseña encriptada/hasheada: contraseña del usuario. Si no se especifica, el usuario no podrá acceder al sistema.
- m: crea el directorio de trabajo si no existe o no se especifica. Se copian los archivos de configuración de /etc/skel.
- s shell: establece un intérprete de comandos al usuario. Por defecto, se emplea /bin/bash.

2. Gestión de usuarios por línea de comandos en Linux

Ejercicio:

1. Crea un usuario “maria” con grupo principal maria, estableciéndose su directorio en /home/maria y copiando en él los archivos de configuración por defecto de /etc/skel.

   $ sudo useradd -d /home/maria -m maria

2. Crea un usuario “german” con grupo principal “games” (ya existente), asignándole como directorio de trabajo /home/German donde se copian los archivos de configuración por defecto de /etc/skel. Además, se asigna a este usuario el shell /bin/bash

2. Gestión de usuarios por línea de comandos en Linux

Modificar usuarios

usermod [-c comentario] [-g grupo] [-G grupo[, grupo …]] [-d directorio_trabajo [-m]] [-p contraseña_encriptada] [-e fecha] [-f dias] [-l nuevoLogin] [-L] [-U] [-s shell] usuario

- c comentario: establece valores asociados al quinto campo del fichero /etc/passwd.
- d directorio: asigna un nuevo directorio de trabajo. Si se emplea junto con el modificador “–m”, todo el contenido del antiguo directorio de trabajo se moverá al nuevo.
- g grupo: asignación al nuevo grupo principal.
- G grupos: lista de grupos secundarios separados por comas y sin espacios. Se eliminará su asociación con los grupos secundarios anteriores, a menos que se indique con el modificador “-a” que se añaden los grupos nuevos a los anteriores.
- l nuevoLogin: se modifica el login anterior al nuevo login aportado.
- s shell: se modifica el anterior shell al nuevo shell aportado.

2. Gestión de usuarios por línea de comandos en Linux

Modificar usuarios

Los usuarios disponen de información adicional que se almacena en el quinto campo del fichero /etc/passwd.

Dependiendo de la distribución de GNU/Linux, esta información puede variar. Para actualizarla, se emplea el comando:

chfn

2. Gestión de usuarios por línea de comandos en Linux

Eliminar usuarios

userdel [-r] login

- Donde el modicador “–r” permite eliminar la carpeta home del usuario.

2. Gestión de usuarios por línea de comandos en Linux

Ejercicio:

1. Crea dos usuarios con los archivos de configuración del /etc/skel. Elimina un usuario, manteniendo sus archivos. Elimina totalmente el otro usuario. Comprueba las acciones.

   # userdel maria

   # userdel -r german

2. Muestra el contenido del directorio /etc/skel, mediante ls –la /etc/skel. Presta atención a cada fichero

2. Gestión de usuarios por línea de comandos en Linux

Comprobar usuarios conectados

$ who [am i] [-u] [-H] [-q]

- am i: muestra el usuario actual. Esto tiene mucho sentido, ya que podemos trabajar en varios terminales, o cambiando de identidad (mediante su, por ejemplo), llegando a perder la noción del usuario con el que estamos actuando en un preciso momento.
- u: muestra información de los usuarios conectados: login, terminal, fecha y hora de conexión, tiempo de inactividad e identificador del proceso shell de usuario.
- H: imprime cabeceras.
- q: muestra solamente los logins y el número de usuarios conectados.

2. Gestión de usuarios por línea de comandos en Linux

Crear grupos

groupadd [-g GID] nombre_grupo

- Donde: g GID: asigna un identificador de grupo al nombre de grupo. Por defecto, hemos de indicar un valor igual o superior a 1000, y no se deben repetir.

Eliminar grupos

groupdel nombre_grupo

- Deben de eliminarse primero los usuarios del grupo.

2. Gestión de usuarios por línea de comandos en Linux

Modificar grupos

groupmod [-g GID] [-n nuevo_nombre_grupo] nombre_grupo

- g GID: indica el nuevo identificador de grupo. Al realizar esta modificación, hemos de prestar atención, puesto que los ficheros con el antiguo GID deben ser asignados manualmente al nuevo GID.
- n nuevo_nombre_grupo: especifica el nuevo nombre para el grupo.

Comprobar grupos de un usuario

$ groups [usuario] . Al ejecutar este comando, el primer grupo es el principal y los sucesivos son los secundarios.

$ id [usuario] Muestra la información más detallada, especificando, además, el UID y los GID de los grupos

2. Gestión de usuarios por línea de comandos en Linux

Añadir usuario a grupo

adduser [login_usuario] grupo

Eliminar usuario de grupo

#deluser [login_usuario] grupo

2. Gestión de usuarios por línea de comandos en Linux

Grupos predeterminados

- adm Grupo de administración que permite accesos a archivos de registro y comandos como sudo y su
- users Grupo de usuarios estándar
- nobody Sin privilegios
- root Administración sin restricciones sobre todo el sistema
- tty Aporta privilegios sobre algunos dispositivos, como /dev/tty

2. Gestión de usuarios por línea de comandos en Linux

Ejercicio

1. Crea un grupo llamado informatica2
2. Crea en entorno 3 usuarios: ssii1401, ssii1402, ssii1403, asignándole carpeta usuario /home/nombre, Shell, id de usuario: 1401, 1402, 1403
3. Asigna el grupo informática2 a los tres usuarios anteriores.
4. Modifica la contraseña de cada usuario a login_usuario1234
5. Identifícate como ssii1401 y crea en tu carpeta un archivo prueba-ssii1401.txt con contenido.
6. Haz que ssii1402 añada una línea al fichero anterior.

2. Gestión de usuarios por línea de comandos en Linux

Cambio propietario fichero

chown [-R][-h] nuevo_propietario[.nuevo_grupo] fichero

- R: modo recursivo para aplicar los cambios al contenido de directorios.
- h: afecta al enlace simbólico, en lugar del archivo referenciado, ya que sin esta opción solo afectaría al archivo referenciado.

El propietario del archivo se sustituirá por nuevo_propietario y el grupo por nuevo_grupo.

El separador entre el propietario y el grupo es “.” o “:”, dependiendo de la distribución de Linux, aunque en sistemas GNU lo más común es “.”

2. Gestión de usuarios por línea de comandos en Linux

Cambio grupo fichero

chgrp [-R] nuevo_grupo ficheros

- R: modo recursivo para aplicar los cambios al contenido de directorios.

Para cambiar un fichero de usuario o de grupo se puede hacer con sudo o con un usuario que tenga permisos del usuario/grupo origen como del usuario/grupo destino.

2. Gestión de usuarios por línea de comandos en Linux

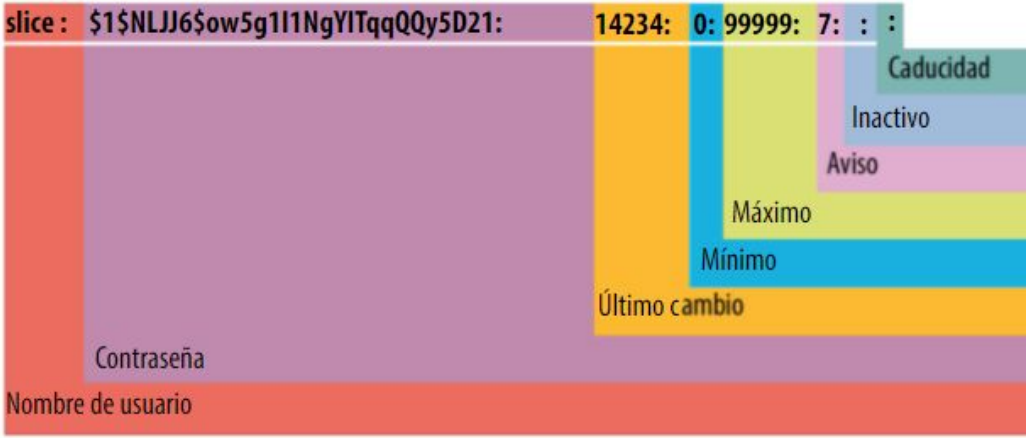
Ejercicio

1. Cómo cambiarías al fichero prueba-ssii1401.txt para que su grupo propietario sea informática2? Intenta el cambio y explica cómo lo has hecho.
2. ¿Puede ssii1402 añadir ahora una línea al fichero anterior? ¿Por qué?

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

La seguridad de las cuentas de usuarios se basa en las contraseñas, y estas se gestionan gracias al fichero de configuración /etc/shadow, del cual Linux-PAM hace uso, y que contiene las cuentas de los usuarios del sistema. Cada fila se corresponde con un usuario (al igual que /etc/passwd):



2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

1. Login de usuario.
2. Password encriptada/hasheada. Dependiendo del algoritmo de cifrado, puede variar la extensión de

la contraseña encriptada. Su comienzo especifica este: 1 (MD5), 5 (SHA-256), 6 (SHA-512), etc. Si la contraseña comienza por “!” indica que se encuentra bloqueada.

3. Días transcurridos desde el 1/1/1970, cuando la contraseña fue cambiada por última vez.
4. Número de días, como mínimo, que han de pasar para que el usuario pueda cambiar la contraseña.
5. Número máximo de días que la contraseña es válida.
6. Número de días que el sistema avisa antes de que caduque la contraseña

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

Para cambiar o añadir una contraseña a un usuario tenemos el comando:

$ passwd [usuario]

Si no indicamos el login del usuario la cambia para el usuario que ejecuta el comando.

Para poder usar la opción -p al crear un usuario con useradd podemos usar una función criptográfica de Linux, openssl.

$ openssl passwd [opciones] contraseña

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

Entre las opciones de openssl passwd tenemos:

-1 hash MD5

-5 hash SHA256

-6 hash SHA512

-salt cadena aleatoria

Se puede combinar generar la contraseña con la creación del usuario con:

$ sudo useradd -p $(openssl passwd -6 1234) usuario

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

Ejercicio:

Calcula el hash de tu contraseña para SHA512, compáralo con el del fichero /etc/shadow.

Obten tu hash de /etc/shadow.

Descarga las 10k-most-common.txt de

Common-Credentials

Calcula los hashes de las 10k contraseñas más comunes con la sal de tu contraseña, guárdalo en un fichero llamado pre-hash.txt .

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas - Ataques de diccionario

Como acabamos de ver es recomendable que la contraseña no esté en diccionarios, la mayor recopilación hasta la fecha de hoy:

> https://hipertextual.com/2021/06/rockyou2021-la-mayor-filtracion-de-contrasenas-de-la-historia-revela-8-400-millones-de-ellas
>
> Como mínimo deberíamos comprobar que nuestras contraseñas no están entre las 10M más comunes que salen aquí Common-Credentials

Que la contraseña tenga sal es básico para evitar que alguien pueda usar diccionarios pre-hasheados.

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas-ataques fuerza bruta

Además de los ataques de diccionario tenemos que evitar que sea debil a ataques de fuerza bruta: Ataque de fuerza bruta - Wikipedia, la enciclopedia libre

Tenemos la guía del CCN que nos proporciona recomendaciones:

Guía de Seguridad de las TIC CCN-STIC 821

También tenemos la oficina de seguridad del internauta.

Aprende a gestionar tus contraseñas | Oficina de Seguridad del Internauta

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas-ataques fuerza bruta

Como programadores web, debemos exigir a los usuarios que pongan contraseñas seguras.

Realiza un resumen del punto 7 de la guía del CCN.

¿Cuántos caracteres debe tener como mínimo la contraseña?

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas-ataques fuerza bruta

Las recomendaciones están bien pero vamos a entender cómo funciona:

La incertidumbre se mide en entropía->

$$\mathrm{H}(X) = -\sum_{i=1}^{n} \mathrm{P}(x_i) \log \mathrm{P}(x_i)$$

Para comparar lo aleatoria que es una contraseña con respecto a otra usamos una unidad común bits de entropía **H**->

Y por último la fórmula para calcularla:

N el número de símbolos posibles

L la longitud de la contraseña

$$H = \log_2 N^L = L \log_2 N = L \frac{\log N}{\log 2}$$

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas-ataques fuerza bruta

Ejercicios:

Calcula la entropía de las siguientes contraseñas:

- 10 símbolos siendo números aleatorios.
- 8 símbolos siendo letras en minúscula aleatorias.
- 6 símbolos siendo letras en mayúscula y minúscula aleatorias.
- 6 símbolos siendo letras en mayúscula y minúscula y números aleatorios.
- 8 símbolos siendo letras en mayúscula y minúscula y números aleatorios.
- 8 símbolos siendo caracteres de tu teclado aleatorios.

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas-ataques fuerza bruta

Ejercicios:

- L'ANSSI l'Autorité Nationale en matière de Sécurité et de défense des systèmes d'Information, un equivalente al CCN de España clasifica las contraseñas según una tabla. ¿Calcula la longitud de una contraseña fuerte si usamos los 95 caracteres imprimibles de ASCII, los 218 caracteres imprimibles de ASCII extendido?

Taille de clé équivalente	Force d'un mot de passe
64	très faible
64<80	faible
80<100	moyen
> 100	fort

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas-ataques fuerza bruta

Forma de generar una contraseña fuerte con Diceware:

The Diceware Passphrase Home Page

Esta es la mejor forma de crear una contraseña segura – Blog EHCGroup.

Gestores de contraseñas, son una buena opción para conseguir que las contraseñas sean únicas para cada servicio:

Open source: keepass y bitwarden

Privativo: Lastpass

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas-ataques fuerza bruta

Forma de generar una contraseña fuerte con Diceware:

The Diceware Passphrase Home Page

Esta es la mejor forma de crear una contraseña segura – Blog EHCGroup.

Gestores de contraseñas, son una buena opción para conseguir que las contraseñas sean únicas para cada servicio:

Open source: keepass y bitwarden

Privativo: Lastpass

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

En Linux se pueden gestionar y establecer políticas de caducidad de contraseñas mediante los comandos passwd y chage, modificando así valores del archivo de congufiración /etc/shadow.

chage [opciones] [login]

$ chage -l usuario -> podemos ver la lista de opciones para la contraseña del usuario

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

Opciones de chage:

- -d o –lastday fecha/días: establece la fecha o el número de días desde el 1 enero de 1970 desde que se cambió la contraseña, con 0 obliga a cambiar contraseña:

  $sudo chage -d 10 usuario

  $chage -l usuario

Las fechas en formato AAAA-MM-DD

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

Opciones de chage:

- -E o --expiredate fecha: establece la fecha (en número de días desde 1/1/1970) a partir de la cual la cuenta del usuario caduca y, por tanto, no será accesible. A -1 indica que no existe tal limitación y a 1 la cuenta se deshabilita de inmediato.

  Las fechas en formato AAAA-MM-DD

- -I o --inactive días: establece el número de días de inactividad después de que una contraseña haya expirado antes de que la cuenta se bloquee. Para poder desbloquear la cuenta, el usuario deberá ponerse en contacto con un administrador o el usuario root. A -1 indica que no existe tal limitación y a 0 se deshabilitará la cuenta en cuanto expire la contraseña.

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

Opciones de chage:

- -l o --list: muestra la información de caducidad de la contraseña.
- -m o --mindays días: establece el número mínimo de días para poder cambiar la contraseña. A 0 indica que no existe tal limitación.
- M o --maxdays días: establece el número máximo de días para poder cambiar la contraseña, es decir, para que caduque, desde que se cambió la contraseña por última vez. Transcurrido este número máximo de días, el sistema instará al usuario para que cambie la contraseña, antes de poder usar su cuenta. A -1 indica que no existe tal limitación.
- -W o --warndays días: establece el número de días de aviso previos a que la contraseña caduque

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

Al bloquear o dejar en blanco una contraseña, no se llega a deshabilitar la cuenta del usuario. Para ello, hemos de emplear chage y usermod (con la opción “- - expiredate 1”). De lo contrario, el usuario podría acceder al sistema por otros medios, como, por ejemplo, ssh.

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

passwd, además de establecer la contraseña para un usuario, permite modificar otros valores relativos a esta. Opciones

- -d o -- delete: deja en blanco la contraseña del usuario.
- -e o -- expire: hace expirar la contraseña de usuario, haciendo que en la siguiente conexión el sistema la solicite.
- -i o -- inactive DIAS: establece el número de días de inactividad después de que una contraseña haya expirado, antes de que la cuenta se bloquee. A -1 indica que no existe tal limitación y a 0 se deshabilitará la cuenta en cuanto expire la contraseña.

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

passwd, además de establecer la contraseña para un usuario, permite modificar otros valores relativos a esta. Opciones

- -l o -- lock: bloquea la contraseña de la cuenta de usuario.
- -u o -- unlock: desbloquea la contraseña de la cuenta de usuario.
- -x o -- maxdays días: establece el número máximo de días para poder cambiar la contraseña, es decir, para que caduque, desde que se cambió la contraseña por última vez. Transcurrido este número máximo de días, el sistema instará al usuario para que cambie la contraseña, antes de poder usar su cuenta. A -1 indica que no existe tal limitación.
- -w o -- warndays días: establece el número de días de aviso previos a que la contraseña caduque.

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

usermod dispone de varias opciones relacionados con la seguridad de la cuenta o la contraseña de los usuarios:

- -e o -- expiredate fecha
- -f o -- inactive días
- -L o -- lock: bloquea la contraseña de la cuenta de usuario.
- -U o -- unlock: desbloquea la contraseña de la cuenta de usuario.
- -p o -- password CONTRASEÑA ENCRIPTADA: asigna una contraseña ya encriptada a un usuario.

2. Gestión de usuarios por línea de comandos en Linux

Seguridad de cuentas de usuario y contraseñas

Ejercicio. Para los usuarios usuario0, usuario1, usuario2, usuario3 y usuario4:

1. Suministra la contraseña sistEmas_%20 a todos los usuarios.
2. Deshabilita la cuenta de usuario3 y bloquea la contraseña de usuario4.
3. Habilita la cuenta de usuario3 y desbloquea la contraseña de usuario4.
4. Establece 20 días como máximo y 10 días como mínimo para cambiar la contraseña de usuario2.
5. Establece una fecha para usuario4, a partir de la cual la cuenta caducará y será inaccesible. Compruébalo. Elimina la expiración de la cuenta y compruébalo.

2. Gestión de usuarios por línea de comandos en Linux

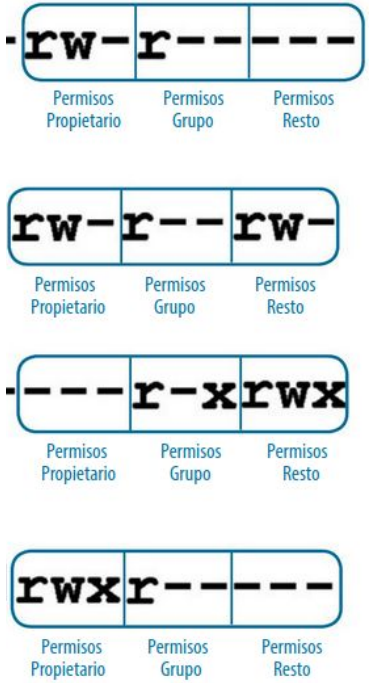
Recursos y permisos locales

La información de permisos guardada en i-nodo de cada archivo o directorio es la siguiente:

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Ejemplos de permisos:



2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Los bits especiales pueden tomar 3 valores:

- Set-uid: si está activo el usuario que ejecuta el archivo toma los permisos del propietario. Aparece una s o S en los permisos del propietario en lugar de una x. Para buscarlos ejecutar el siguiente comando:

  $ find / -perm -4000

¿Qué archivos te suenan?

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Los bits especiales pueden tomar 3 valores:

- Set-gid: si está activo en un fichero el usuario que ejecuta el archivo toma los permisos del grupo. Si está activo en un directorio todos los archivos dentro del directorio pertenecerán al grupo del directorio. Aparece una s o S en los permisos del grupo en lugar de una x. Para buscarlos ejecutar el siguiente comando:

  $ find / -perm -2000

¿Qué archivos te suenan?

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Los bits especiales pueden tomar 3 valores:

- Sticky-bit: solo afecta a directorios. Si está activo fuerza que no puedan modificar ni eliminar los ficheros del directorio el resto de usuarios. Es un mecanismo de seguridad. Aparece una t o T en los permisos del resto de usuarios en lugar de una x. Para buscarlos ejecutar el siguiente comando:

  $ find / -perm -1000

¿Qué directorios te suenan?

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Ejercicio:

Ejecuta ls –l /usr/bin/passwd. Observa quién es el propietario y si se encuentra activo el set-uid en dicho archivo.

Ejecuta ls –ld /tmp. Observa quién es el propietario, el grupo y si se encuentra activo el sticky-bit del directorio.

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Con chmod se modifican los permisos de los ficheros:

$ chmod [-R] permisos ficheros

Con -R lo hace recursivo a todos los ficheros de la carpeta.

Los permisos se pueden modificar en [octal/o en decimal de 3 en 3 bits] poniendo 1 o 0 en el bit que queremos activar:

rwx rw- r-- -> 111 110 100 -> 7 6 4

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Además, para habilitar los modos especiales sobre la máscara de permisos hemos de aplicar el siguiente procedimiento:

- Sticky-bit: sumar 1000 en octal al resultado anterior.
- Set-gid: sumar 2000 en octal al resultado.
- Set-uid: sumar 4000 en octal al resultado anterior.

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Ejercicio:

Crear un archivo prueba.txt y aplicar las siguientes máscaras de permisos en octal/decimal, sobre prueba.txt, comprobando su resultado: “rwsrwxrwT”, “rwxrwxrw-“ y “rw-r-srw-“

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Otra opción es usar el modo simbólico en lugar de numérico.

chmod modo_simbolico[,modo_simbolico] archivos

El modo simbólico se puede repetir más de una vez, separándose por comas. Su formato es:

Destino	Operación	Permisos
[u][g][o][a]	{+ = -}	[r] [w] [x] [s] [t]

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Destinatarios de los permisos para modificar:

“u”: propietario.

“g”: grupo.

“o”: otros (el resto de usuarios).

“a”: propietario, grupo y otros.

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Tipo de modificación:

“+”: se añaden al valor actual.

“=”: establece los permisos especificados y anula el resto.

“-“: se quitan al valor actual.

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Permisos:

“r”: lectura.

“w”: escritura.

“x”: ejecución.

“s”: set-uid o set-gid, dependiendo de su aplicación al propietario o al grupo.

“t”: sticky-bit.

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Ejemplo

chmod go-r prueba.txt

¿Qué hace?

Sobre el archivo prueba.txt, aplicar las siguientes modificaciones sobre la máscara de permisos comprobando sus resultados.

- Deshabilitar para el grupo y otros el permiso de lectura.
- Habilitar el permiso de ejecución para el propietario y deshabilitar el set-gid.
- Conceder permisos de lectura y escritura para el usuario y el grupo, anulando el resto de permisos. Para el resto de usuarios se mantienen sus permisos.

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Permisos por defecto:

Los permisos originales de un archivo y un directorio son 0666 y 0777 en octal. Además al crear el archivo se usa una máscara de permisos, típicamente 0022, para modificar el resultado final. Para ver o modificar la máscara de permisos, empleamos el comando umask (user mask) con la siguiente sintaxis:

umask [máscara]

Máscara ha de ser el valor de la máscara de permisos en octal (aunque se puede emplear la opción “-S” para usar la notación simbólica). Si no se especifica, mostrará el valor de la máscara actual.

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Permisos por defecto:

El procedimiento de aplicación de la máscara de permisos es el siguiente:

1. Convertimos la máscara de permisos de octal a binario.
2. Aplicamos el operador NOT sobre la máscara en binario.
3. Realizamos la operación AND entre los permisos originales de archivos o directorios y la máscara.

Ejercicio: Obtener los permisos de los archivos y directorios recién creados aplicando una máscara de permisos 0022

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

La máscara se aplica para ficheros creados por terminal. Se puede hacer para todos los archivos modificando la variable en /etc/profile o ~/.bashrc

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

Archivos de configuración:

Tipos	Archivos	Descripción
Archivos globales	/etc/skel	Directorio que contiene la plantilla de creación de perfiles de usuarios.
	/etc/profile	Configuración genérica de perfiles cuando se inicia sesión en el sistema como *login shell*.
	/etc/bash.bashrc	Configuración genérica de perfiles cuando se inicia sesión con Shell Bash *interactivo* (ya sea *login* o *non-login*).
Archivos locales	~/.bashrc	Configuración local de usuario que se ejecuta cuando este inicia sesión con Shell Bash como *non-login shell*.
	~/.bash_logout	Configuración local de usuario que se ejecuta cuando este termina sesión con Shell Bash
	~/.profile	Configuración local de usuario cuando este inicia sesión en el sistema con un shell de inicio de sesión como *login shell*.

2. Gestión de usuarios por línea de comandos en Linux

Recursos y permisos locales

- Login shell: aquel que cuando se inicia el shell autentica al usuario solicitando usuario y contraseña. Como, por ejemplo, cuando abrimos un nuevo terminal virtual (CTRL+ALT+F1).
- Non-login shell: aquel que no solicita autenticación del usuario, como, por ejemplo, cuando abrimos un terminal en Gnome o cuando lanzamos el comando bash dentro de un login shell.
- Shell interactivo: aquel que permite interactuar con el terminal escribiendo comandos, interrumpirlos, etc., al estar asociado a un terminal. Ejemplos de ello son los terminales virtuales o los terminales Gnome.
- Shell no interactivo: no está asociado a un terminal, como suele ser un subshell, que normalmente se ejecuta en procesos automáticos, como, por ejemplo, cuando se ejecuta un script.

Como podemos ver en el fichero ~/.bashrc, tenemos variables y alias de comandos.

3. Gestión de usuarios por interfaz gráfica en Windows

La creación de usuarios se puede hacer desde:

Configuración -> Cuentas -> Otros usuarios -> Agregar otra persona a este equipo.

Panel de control -> Cuentas de usuario -> Administrar otra cuenta

3. Gestión de usuarios por interfaz gráfica en Windows

Usuarios y grupos locales

Las opciones avanzadas para administrar usuarios y grupos se realiza desde:

Panel de control -> Herramientas administrativas -> Administración de equipos -> Usuarios y grupos locales

Click derecho, propiedades. Se pueden ver las propiedades de los usuarios. ¿Qué opciones están disponibles?

Hay unas cuentas de usuario y grupos administrativos creadas por Windows.

3. Gestión de usuarios por interfaz gráfica en Windows

Configuración de las contraseñas en Windows

Para establecer los requisitos que deben cumplir las contraseñas, en Windows, se accede al Editor de directivas de grupo local (búsqueda en inicio o ejecutar gpedit.msc).

Una vez dentro: Configuración de equipo -> Configuración de Windows -> Directivas de cuenta -> Directiva de contraseñas y Directiva de bloqueo

¿Qué podemos definir?

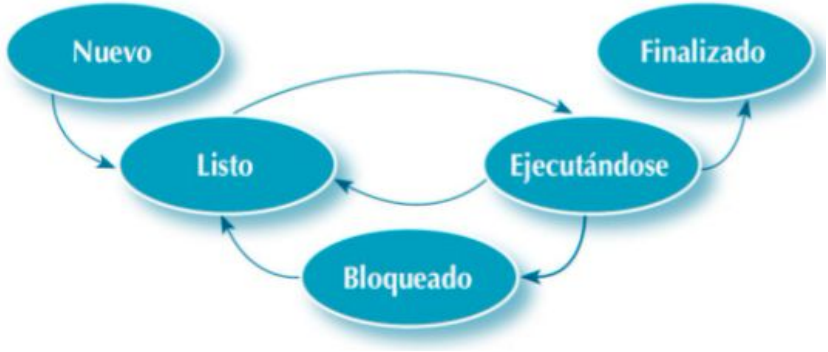
4. Gestión de procesos por línea de comandos en Linux

Las instancias de los programas en ejecución, también llamados tareas o procesos, son administrados por el sistema operativo como un recurso más.

El planificador de procesos del sistema operativo gestiona todos los procesos mediante operaciones de creación, comunicación, compartición y finalización de procesos.

4. Gestión de procesos por línea de comandos en Linux

Como vimos en la UD2, al crearse un nuevo proceso se le asigna una zona de memoria y se guarda la información del mismo en el bloque control de procesos. Siguiendo el proceso inverso al finalizarse.



4. Gestión de procesos por línea de comandos en Linux

Los procesos disponen de un identificador único llamado PID (IDentificador de Proceso). El PCB de cada proceso almacena información esencial, como:

- Identificación del proceso (PID).
- Identificación del proceso padre (PPID), es decir, el PID del proceso que lo creó.
- Usuario propietario.
- Valores del estado del procesador en el momento de producirse el cambio de contexto.
- Estado.
- Valores de referencia de memoria RAM.
- Ficheros abiertos.
- Buffers de memoria utilizados

4. Gestión de procesos por línea de comandos en Linux

Con el comando ps obtenemos la información de los procesos:

$ ps [opciones]

Las opciones más comunes son:

Obtener información de todos los procesos del sistema

$ ps aux

$ ps aux -- sort cputime (se ordenan por tiempo consumido de CPU)

Imprimir información junto a un árbol de procesos

$ ps axjf

4. Gestión de procesos por línea de comandos en Linux

Seleccionar procesos por usuario, PID, terminal, etc.

$ ps –U usuario –u usuario u (muestra los procesos del usuario luis)

$ ps -- pid 8943 (información del proceso con PID 8943)

$ ps –t /dev/pts/2 (información de los procesos del terminal virtual “/dev/pts/2”)

4. Gestión de procesos por línea de comandos en Linux

Información del comando ps:

1. UID del usuario propietario del proceso.
2. PID del proceso.
3. PPID del proceso.
4. Índice de utilización reciente del procesador (columna titulada “C” y empleada para calcular la prioridad entre procesos).
5. Tiempo de inicio del proceso (Start Time o STIME).
6. Terminal de lanzamiento (TTY).
7. Tiempo de CPU consumido (TIME).
8. Orden de ejecución (COMMAND).
9. Tamaño del proceso en la memoria virtual en KB (VSZ).
10. Tamaño de la memoria residente del proceso (en memoria física) en KB (RSS).
11. Porcentaje de uso de CPU (%CPU).
12. Porcentaje de uso de memoria (%MEM).
13. Estado del procesador (STAT o S).

4. Gestión de procesos por línea de comandos en Linux

Estado del procesador STAT o S:

Estado	Descripción
R	Ejecutándose o listo para ser ejecutado (Runnable)
S	Bloqueado o durmiendo (Sleeping)
T	Parado (Trace)
Z	Zombi (proceso que ha muerto, pero el proceso padre no ha reconocido su muerte)
I	Inactivo en creación (Idle)
N	Con prioridad menor de lo normal (NICE)
<	Con prioridad mayor de lo normal
+	Se encuentra en el grupo de procesos en primer plano
s	Proceso líder de sesión
l	Es un proceso multihilo (un mismo proceso con diferentes tareas que se pueden ejecutar en paralelo, evitando así el cambio de contexto)

4. Gestión de procesos por línea de comandos en Linux

Un uso muy extendido es filtrar el resultado de ps, concatenando el comando grep.

```
$ ps aux | grep bash
```

Para mostrar la estructura arborescente de procesos podemos utilizar el comando pstree.

```
$ pstree
```

4. Gestión de procesos por línea de comandos en Linux

El comando top muestra la información del sistema y los procesos de manera continua (no una foto fija com ps). Información de top:

- Línea 1.ª: hora actual, tiempo del sistema encendido, número de usuarios y carga media en intervalos de 1, 5 y 15 minutos, respectivamente.
- Línea 2.ª: número de tareas, número de procesos en estado, ejecutándose o listos (R), bloqueados o hibernando (S), parados (T) y zombis (Z) respectivamente.
- Línea 3.ª: tiempos de CPU de usuario, del kernel del sistema, etc.
- Línea 4.ª: tamaño en MB de memoria física en total, libre, usada y utilizada por buffer.
- Línea 5.ª: tamaño en MB de memoria virtual total, libre usada y disponible.

4. Gestión de procesos por línea de comandos en Linux

El resto de líneas muestran la información de los procesos, identificando las columnas mediante cabeceras similares a las del comando ps.

Podemos modificar el tiempo de actualización de la información mostrada en segundos (por defecto, tres segundos), monitorizar determinados procesos por PID o usuarios:

- Mostrar los procesos del usuario luis actualizando la información cada dos segundos: $ top –d 2 –u usuario
- Mostrar información del proceso con PID 8933: $ top –p 8933

4. Gestión de procesos por línea de comandos en Linux

Procesos en primer y segundo plano

Cuando ejecutamos un proceso en terminal no se puede realizar ninguna acción hasta que este termina. El proceso se ejecuta en primer plano.

Existe una alternativa para evitar que el propio usuario tenga que esperar a la terminación de una tarea para poder continuar con la ejecución de otras nuevas, denominada ejecución en segundo plano o background. Consiste en añadir al nal de la línea de mandatos que ejecutar en el shell, el símbolo &.

Ejemplo:

$ yes > /dev/null para detener proceso pulsar q o crtl +c

$ yes > /dev/null &

4. Gestión de procesos por línea de comandos en Linux

Procesos en primer y segundo plano

Con el comando jobs podemos identificar las tareas que se hallan en segundo plano.

La tarea más reciente con el símbolo “+” y el segundo más reciente con “–“.

Se pueden pasar procesos de primer a segundo plano, y viceversa, mediante los siguientes comandos:

fg [%][numtarea] pasa una tarea a primer plano. Sin argumentos, la tarea más reciente.

$ fg %2

bg [%] [numtarea] pasa una tarea a segundo plano. Sin argumentos, la tarea más reciente.

Para pasar una tarea de primer plano a segundo plano, hemos de detenerla mediante la combinación de teclas CTRL+Z y, a continuación, lanzarla en background con bg.

4. Gestión de procesos por línea de comandos en Linux

Ejercicio:

Empleando el comando sleep: sleep, suspende la devolución del prompt del shell durante 50 segundos.

Pasa esa tarea a segundo plano y después a primer plano, antes de que finalice.

4. Gestión de procesos por línea de comandos en Linux

Prioridad de procesos:

El algoritmo de planificación de Linux que determina el orden de ejecución entre los procesos en la cola de listos, emplea una mezcla de algoritmos como Round Robin, FIFO, prioridades, etc.

La prioridad real de cada proceso se calcula de manera compleja mediante varios factores, la cual se puede alterar relativamente mediante un índice denominado nice.

El valor nice de un proceso oscila entre -20 (máxima prioridad) y 19 (menor prioridad), y puede ser modificada por el propietario del proceso o el superusuario.

4. Gestión de procesos por línea de comandos en Linux

Prioridad de procesos:

Por defecto, un usuario solo puede disminuir la prioridad de sus procesos cuando los lanza.

Cuando se ejecuta un proceso, este lo hace predeterminadamente con una prioridad relativa 0 (valor nice). Podemos comprobarlo con la orden ps –l, donde lo indican las columnas NI (NIce) y PRI (por defecto con valor 80).

$ nice –n+1 yes > /dev/null &

$ ps -l

4. Gestión de procesos por línea de comandos en Linux

Prioridad de procesos:

Una vez ejecutado un proceso se puede modificar su prioridad con renice:

renice prioridad [[-p] pid ...] [[-g] pgrp ...] [[-u] usuario ...]

Para un proceso:

```
$ renice +15 785
```

Para un usuario

```
# renice +20 -u usuario
```

4. Gestión de procesos por línea de comandos en Linux

Envío de señales a procesos

Los procesos reciben señales para ser controlados desde el propio sistema operativo y desde el exterior. Un usuario también puede enviar señales a los procesos mediante la orden kill.

$ kill -señal PID

https://manpages.ubuntu.com/manpages/bionic/es/man7/signal.7.html

4. Gestión de procesos por línea de comandos en Linux

Envío de señales a procesos

Donde las señales más utilizadas son:

- 2 o SIGINT: interrumpe un proceso, similar a CTRL+c. Esta señal puede ser manejada por el propio proceso, aunque no es lo habitual, terminando su ejecución.
- 9 o SIGKILL: mata un proceso.
- 15 o SIGTERM: mata un proceso, aunque esta señal puede ser ignorada en determinados casos, por lo que SIGKILL sería más efectiva.
- 18 o SIGCONT: continúa la ejecución de un proceso.
- 19 o SIGSTOP: pausa la ejecución de un proceso. Similar a CTRL+z

4. Gestión de procesos por línea de comandos en Linux

Envío de señales a procesos

Son muchas las utilidades del envío de señales por parte de los usuarios, como, por ejemplo, detener procesos que consuman muchos recursos, interrumpir procesos que bloqueen otras tareas, matar procesos, etc. La combinación de teclas CTRL+C solo se puede enviar a procesos en primer plano, sin embargo, las señales mediante kill se pueden enviar desde diferentes orígenes, como otros terminales.

4. Gestión de procesos por línea de comandos en Linux

Ejercicio:

Procesos en Linux:

1. Muestra todos los procesos del sistema.
2. Muestra los procesos del usuario actual.
3. Desde un terminal, lanza el proceso yes > /dev/null . Pásalo a segundo plano. Elimina el proceso.
4. Desde un terminal, lanza el proceso yes > /dev/null. Páralo con ctrl-z. Desde otro terminal, localiza el proceso. Comprueba su estado tras el envío la señal.
5. Lanza el proceso sleep 500 en segundo plano y con mínima prioridad. Intenta subir la prioridad. Sube la prioridad al máximo del proceso con root. Disminuye la prioridad del proceso. Comprueba su prioridad tras cada modificación.

4. Gestión de procesos por línea de comandos en Linux

Ejercicio:

Prácticas de LINUX - Comandos básicos: Ejercicios y correcciones (2ª edición) - Enunciado 6.4 Manipulación de procesos

5. Gestión de procesos por interfaz gráfica en Windows

En Microsoft Windows la planificación de procesos se basa en la utilización de colas múltiples por prioridades. Estas colas emplean diferentes algoritmos de planificación; uno de ellos es Round-Robin.

En Windows tenemos el Administrador de tareas. En la pestaña ‘Procesos’, podemos analizar todos los procesos en ejecución del sistema agrupados por:

- Aplicaciones
- Procesos en segundo plano
- Procesos de Windows

En la pestaña procesos más detalles se pueden ver todos los procesos del sistema.

En la pestaña detalles sale más información y se le pueden asignar prioridades como en Linux.

6. Automatización de tareas en Linux

Tenemos distintos comandos para automatizar tareas:

$ at hora [fecha] [-f fichero] -> Se programa y se ejecuta 1 sola vez.

Al ejecutar el comando con una hora nos sale un terminal de at para introducir comandos. Para terminar se pulsa ctrl+d

```
david@personal:~$ at 20:31
warning: commands will be executed using /bin/sh
at> echo Hola mundo
at> <EOT>
job 2 at Sun Apr 17 20:31:00 2022
```

Se puede añadir una fecha de modo opcional, si no se añade se hará la próxima vez que se llegue a la hora. Los comandos a ejecutar se pueden incluir en un fichero con -f, en lugar de teclearlos.

6. Automatización de tareas en Linux

Con batch programamos tareas que se ejecutarán cuando el procesador esté disponible:

$ batch < fichero

Ejecuta los comandos de fichero.

Al igual que at, solo se ejecutan una vez.

6. Automatización de tareas en Linux

Se pueden ejecutar tareas de forma recurrente con cron. Cron es un demonio que lanza scripts definidos en un archivo crontab. El fichero /etc/crontab tiene las de root,

```
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
```

6. Automatización de tareas en Linux

Cada usuario puede tener su propio archivo crontab.

Para la planificación recurrente con cron tenemos que:

- Comprobar si cron está instalado mediante dpkg –l cron.
- Verificar la actividad del demonio cron mediante: systemctl status cron. En caso negativo, podemos iniciarlo mediante systemctl start cron.
- Reiniciar el demonio mediante systemctl restart cron.

Los cheros crontab han de especificar las órdenes y su periodicidad mediante los siguientes valores separados por espacios:

Minutos [0..59] Hora [0..23] Día [1..31] Mes [1..12] Día de la semana [0..6] orden

Para ignorar un campo se usa *. Una lista de valores separados por coma o mediante sus valores mínimos y máximos separados por - , y también mediante un valor de inicio y un valor incremental, separados por / .

6. Automatización de tareas en Linux

Ejemplos:

- 03 * * * * orden Ejecuta orden en el minuto 3 de cada hora y cada día de todos los meses y para todos los días de la semana.
- 00 * * * 0 orden Ejecuta orden cada hora en punto, de todos los días del mes que sean domingos.
- 3/3 2/3 4 4 4 orden Ejecuta orden cada tres minutos empezando por el minuto 3 de las horas 2,5,8, etc. del día 4 de abril y que sea jueves.
- 30 18 20 1-6 * orden Ejecuta orden a las 18:30 horas del día 20 de cada mes desde enero a junio.

6. Automatización de tareas en Linux

Ejemplos:

- 45 06 1/2 * 1,3,5 orden Ejecuta orden a las 06:45 horas de los días impares de cada mes, siempre que sean lunes, miércoles o viernes.
- */15 10-14 * * 6 orden Ejecuta orden cada 15 minutos de 10 a 14 horas todos los sábados.
- 10,30,45 08 */2 * * orden Ejecuta orden a las 08:10, 08:30 y 08:45 cada dos días y cada mes.

6. Automatización de tareas en Linux

Para editar tareas cron lo mejor es usar

$ crontab -e

Con crontab -l vemos las tareas cron para un usuario y las podemos eliminar con crontab -r

Por último, si preferimos crear el fichero cron propio podemos redirigirlo a crontab

$ crontab < fichero.cron

6. Automatización de tareas en Linux

Ejercicio:

- Crea el archivo de texto planificado.txt dentro de 2 minutos.
- Crea los siguientes archivos de texto:planificado [hora]_[fecha].txt, todos los lunes y jueves a las 15 horas y planificado2 [hora]_[fecha].txt, todos los viernes de los meses pares a las 4 am

Para obtener la hora usamos el comando: $ date +"%T"

Para obtener la fecha usamos el comando: $ date +"%x"

Para que se ejecuten dentro de otro comando usamos $(comando)

7. Monitorización y gestión del sistema. Evaluación de prestaciones

Los administradores del sistema deben obtener información del rendimiento del sistema para así detectar o anteponerse a futuros problemas, a la vez que establecer mejoras en el desempeño del mismo. Conocemos algunos comandos que ofrecen información muy valiosa sobre el rendimiento, top y ps.

El comando uptime muestra la hora del sistema, el tiempo del sistema en activo, el número de usuarios y la carga del sistema en intervalos de 1, 5 y 15 minutos, respectivamente. Equivale a la primera línea de la orden top.

También podemos obtener información sobre el espacio libre y usado de la memoria real y física con free.

7. Monitorización y gestión del sistema. Evaluación de prestaciones

Si queremos más información de la memoria RAM, memoria virtual, intercambios entre memoria RAM y disco, interrupciones y el procesador el comando.

$ vmstat [tiempo [actualizaciones]]

tiempo: es el tiempo en segundos transcurrido entre dos actualizaciones.

actualizaciones: es el número de muestras. vmstat 3 5

También es importante el uso de df, que muestra el porcentaje de uso de las unidades de almacenamiento del sistema.

Con respecto a los usuarios, el comando w muestra quién está conectado y qué está haciendo (parecido a who).

7. Monitorización y gestión del sistema. Evaluación de prestaciones

En Microsoft Windows, a través del ‘Administrador de tareas’, podemos hacer un estudio preciso del rendimiento del sistema en la pestaña ‘Rendimiento’. Podemos analizar el uso de CPU, cantidad de procesos y subprocesos activos, estado de la memoria RAM (en uso, disponible, espacio paginado), usos de discos, tráfico de los adaptadores de red, etc.

También permite analizar los porcentajes de CPU, memoria, disco y red empleados por cada usuario y un desglose por aplicaciones de cada usuario en la pestaña ‘Usuarios’.

Además, podemos estudiar las aplicaciones en segundo plano y los servicios de Windows en las pestañas ‘Detalles’ y ‘Servicios’.

8. Aplicaciones para el mantenimiento y optimización del sistema

Existen aplicaciones propias del sistema operativo o de terceros que aumentan la funcionalidad, mejoran la manejabilidad o incrementan la seguridad del sistema. Podemos clasificarlas en:

a) Aplicaciones de actualización y control de drivers. Aunque el propio sistema operativo se encarga de actualizar los drivers del hardware del sistema, a veces se pueden dar situaciones de incompatibilidad entre una actualización del sistema operativo y un driver, dejando en estado inconsistente el hardware. Por tanto, puede ser adecuado una aplicación que advierta sobre actualizaciones de los drivers del sistema. Algunas de ellas son Drivers Cloud y Driver Booster.

8. Aplicaciones para el mantenimiento y optimización del sistema

b) Aplicaciones de sincronización, copias de seguridad e imágenes del sistema. Existe una gran variedad de aplicaciones que ayudan a automatizar el proceso de copias de seguridad. Muchas de ellas disponen de una larga experiencia y, por tanto, tienen un gran número de seguidores, como EaseUS Todo Backup Free (realizar backups), Clonezilla (gran habilidad para realizar imágenes del sistema), FreeFileSync (sincronizar elementos).

8. Aplicaciones para el mantenimiento y optimización del sistema

c) Antivirus. Es por todos conocida la importancia de disponer de aplicaciones antimalware de terceros, con una gran experiencia en la detección y eliminación de todo tipo de software malicioso. Empresas como Avira, Panda, Bitdefender, Kaspersky, AVG o Avast ofrecen gran variedad de software antispyware, firewall, antimalware, antipop-ups, etc.

d) Optimización del sistema. Una explotación eficiente del sistema operativo implica un mantenimiento cuidado con la descarga de archivos, instalación y desinstalación de programas, control de carpetas temporales, etc. Algunos programas que facilitan estas tareas son CCleaner (Microsoft Windows) y Stacer (Linux)

FIN