1. **Crear** un archivo en **Google Documents o Word en la computadora**
2. **Investigar** y **contestar** las siguientes preguntas.

¿Qué es un usuario root en Linux?

En Linux el usuario root es aquel que tiene todos los permisos en el sistema operativo, es decir, es el súper administrador. Puede acceder a cualquier archivo y también ejecutar cualquier comando, incluidos los que nunca deberías ejecutar.

El usuario root puede hacer lo que quiera en el sistema operativo, así que hay que utilizarlo con mucho cuidado porque podríamos llegar a dejar nuestro sistema inutilizable por un comando mal ejecutado.

¿Por qué ubuntu no me deja establecer la contraseña durante la instalación?

En Linux (y Unix en general), hay un superusuario llamado root. El equivalente de Windows de root es el grupo Administradores. El superusuario puede hacer cualquier cosa, y por lo tanto hacer el trabajo diario como superusuario puede ser peligroso. Podría escribir un comando incorrectamente y destruir el sistema. Idealmente, se ejecuta como un usuario que solo tiene los privilegios necesarios para la tarea en cuestión. En algunos casos, este es necesariamente root, pero la mayoría de las veces es un usuario regular.

De forma predeterminada, la contraseña de la cuenta raíz está bloqueada en Ubuntu. Esto significa que no puede iniciar sesión como root directamente o usar el comando su para convertirse en usuario root. Sin embargo, dado que la cuenta raíz existe físicamente, aún es posible ejecutar programas con privilegios de nivel raíz. Aquí es donde entra sudo: permite a los usuarios autorizados para ejecutar ciertos programas como root sin tener que conocer la contraseña de root.

Esto significa que en la terminal debe usar sudo para los comandos que requieren privilegios de root; simplemente anteponga sudo a todos los comandos que necesite ejecutar como root. Para obtener ejemplos de uso más extensos, consulte a continuación. De manera similar, cuando ejecuta programas GUI que requieren privilegios de root (por ejemplo, el subprograma de configuración de red), use sudo gráfico y también se le solicitará una contraseña (más abajo). Solo recuerde, cuando Sudo solicita una contraseña, necesita SU contraseña de USUARIO y no la contraseña de la cuenta raíz.

Tenga en cuenta que una gran cantidad de usuarios de Ubuntu son nuevos en Linux. Hay una curva de aprendizaje asociada con cualquier sistema operativo y muchos usuarios nuevos intentan tomar atajos habilitando la cuenta raíz, iniciando sesión como raíz y cambiando la propiedad de los archivos del sistema.

¿Cuáles son los procesos típicos de Linux? ¿Cómo identificarlos?

En Linux, un proceso es una instancia de ejecución de un programa o comando. Mientras existan estos procesos, estarán en uno de los cinco estados posibles:

Corriendo o Ejecutable (R)

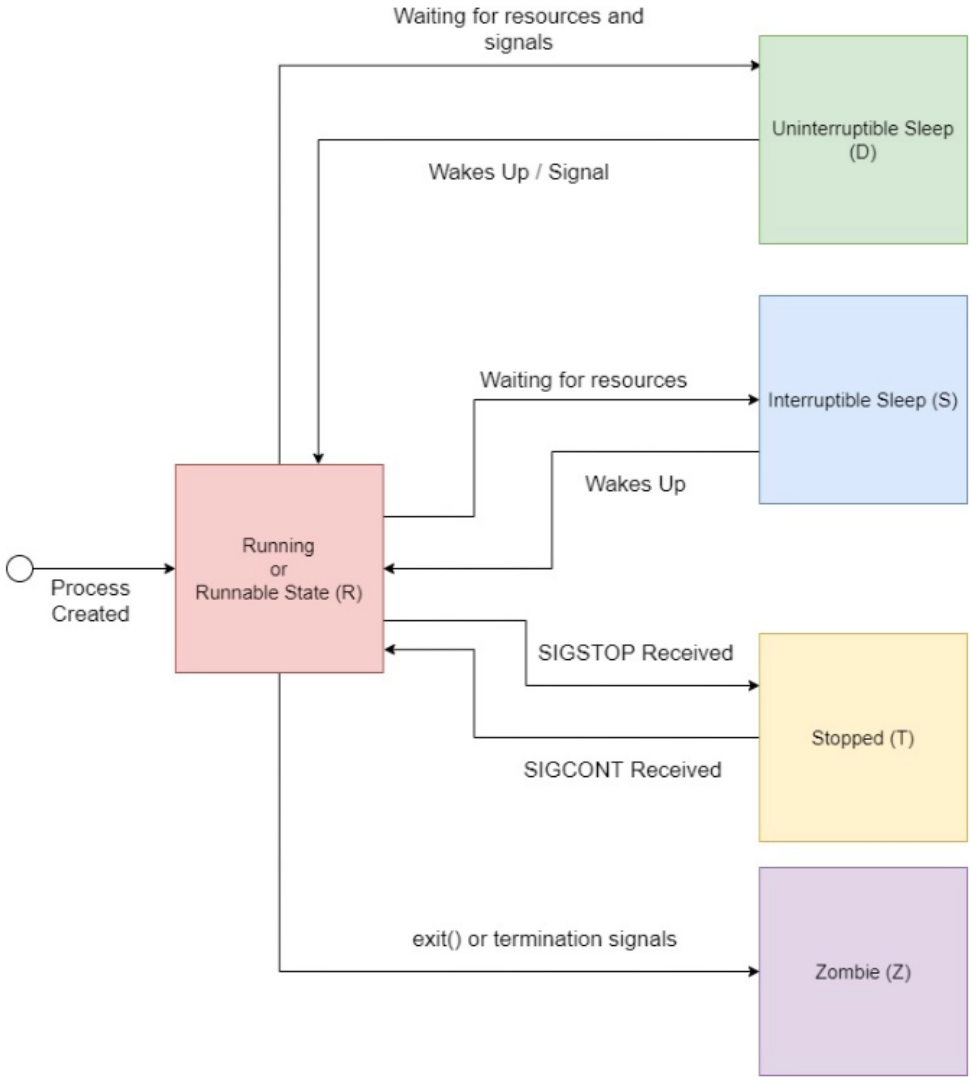
Sueño ininterrumpido (D)

Sueño interrumpible (S)

detenido (T)

Zombi (Z)

Para visualizar el ciclo de vida del proceso, podemos modelarlo en una máquina de estados finitos:



Para cualquier proceso de Linux, su punto de partida es el momento en que se crean. Por ejemplo, un proceso padre puede iniciar un proceso hijo usando la llamada al sistema fork(). Una vez que se inicia, el proceso pasa al estado de ejecución o ejecutable. Mientras se ejecuta el proceso, podría entrar en una ruta de código que requiera que espere recursos o señales particulares antes de continuar. Mientras esperaba los recursos, el proceso abandonaría voluntariamente los ciclos de la CPU al entrar en uno de los dos estados de suspensión.

Además, podríamos suspender un proceso en ejecución y ponerlo en estado detenido. Por lo general, esto se hace enviando la señal SIGSTOP al proceso. Un proceso en este estado seguirá existiendo hasta que se elimine o se reanude con

SIGCONT. Finalmente, el proceso completa su ciclo de vida cuando finaliza y se coloca en un estado zombi hasta que su proceso principal lo borre de la tabla de procesos.

1. Estado de ejecución o ejecutable (R)

Cuando se inicia un nuevo proceso, se colocará en el estado de ejecución o ejecutable. En el estado de ejecución, el proceso ocupa un núcleo de CPU para ejecutar su código y lógica. Sin embargo, el algoritmo de programación de subprocesos podría obligar a un proceso en ejecución a renunciar a su derecho de ejecución. Esto es para garantizar que cada proceso pueda tener una parte justa de los recursos de la CPU. En este caso, el proceso se colocará en una cola de ejecución y su estado ahora es un estado ejecutable esperando su turno para ejecutarse.

Aunque los estados de ejecución y ejecución son distintos, se agrupan colectivamente en un solo estado indicado por el carácter R.

2. Estado durmiente: interrumpible (S) e ininterrumpible (D)

Durante la ejecución del proceso, puede encontrarse con una parte de su código donde necesita solicitar recursos externos. Principalmente, la solicitud de estos recursos se basa en IO, como leer un archivo del disco o realizar una solicitud de red. Dado que el proceso no podría continuar sin los recursos, se estancaría y no haría nada. En eventos como estos, deberían ceder sus ciclos de CPU a otras tareas que están listas para ejecutarse y, por lo tanto, entran en un estado de suspensión.

Hay dos estados de suspensión diferentes: el estado de suspensión ininterrumpida (D) y el estado de suspensión interrumpible (S). El estado de suspensión ininterrumpida solo esperará a que los recursos estén disponibles antes de pasar a un estado ejecutable y no reaccionará a ninguna señal. Por otro lado, el (los) estado (s) durmiente (s) interrumpible (s) reaccionará a las señales y la disponibilidad de recursos.

3. Estado detenido (T)

Desde un estado en ejecución o ejecutable, podríamos poner un proceso en el estado detenido (T) usando las señales SIGSTOP o SIGTSTP. La diferencia entre

ambas señales es que el SIGSTOP que enviamos es programático, como ejecutar kill -STOP {pid}. Además, el proceso no puede ignorar esta señal y pasará al estado detenido. Por otro lado, enviamos la señal SIGTSTP usando el teclado CTRL + Z. A diferencia de SIGSTOP, el proceso puede ignorar opcionalmente esta señal y continuar ejecutándose al recibir SIGTSTP.

Mientras esté en este estado, podríamos devolver el proceso a un estado de ejecución o ejecutable mediante el envío de la señal SIGCONT.

4. Estado zombi (Z)

Cuando un proceso ha completado su ejecución o finaliza, enviará la señal SIGCHLD al proceso principal y pasará al estado zombi. El proceso zombi, también conocido como proceso difunto, permanecerá en este estado hasta que el proceso principal lo elimine de la tabla de procesos. Para borrar el proceso secundario finalizado de la tabla de procesos, el proceso principal debe leer el valor de salida del proceso secundario utilizando las llamadas al sistema wait() o waitpid().

Comprobación del estado del proceso

Hay varias formas de verificar el estado de un proceso en Linux. Por ejemplo, podemos usar herramientas de línea de comandos como ps y top para verificar el estado de los procesos. Alternativamente, podemos consultar el archivo de pseudo estado para un PID en particular.

1. Visualización del estado del proceso mediante ps

Para mostrar el estado del proceso usando ps, ejecutemos el comando ps para incluir una columna que nos diga el estado del proceso:

```
$ ps a
    PID TTY      STAT   TIME COMMAND
   2234 tty2     Ssl+   0:00 /usr/lib/gdm3/gdm-x-session --run-script env GNOME_SHELL_SESSION_MODE=ubuntu
/usr/bin/gnome-session --systemd --session=ubuntu
   2237 tty2     Rl+    0:07 /usr/lib/xorg/Xorg vt2 -displayfd 3 -auth /run/user/1000/gdm/Xauthority -
background none -noreset -keeptty -verbose 3
   2287 tty2     Sl+    0:00 /usr/libexec/gnome-session-binary --systemd --systemd --session=ubuntu
   2982 pts/0    Ss     0:00 bash
   3467 pts/0    R+     0:00 ps a
```

La primera letra del valor debajo de la columna STAT indica el estado en el que se encuentra el proceso. Por ejemplo, el proceso con PID 2234 se encuentra actualmente en un estado de suspensión interrumpible, como lo indica el carácter

S. Además de eso, también podemos observar que el proceso 2237 se encuentra actualmente en estado de ejecución o ejecutable.

Además, podemos ver que hay caracteres adicionales además de cada uno de los caracteres de estado. Estos caracteres indican varios atributos del estado del proceso. Por ejemplo, la letra minúscula s significa que el proceso es el líder de la sesión. Para obtener una lista completa del significado de cada uno de los caracteres, podemos encontrarla en la página de manual oficial.

Usando el comando superior

En Linux, la herramienta de línea de comandos superior muestra los detalles del proceso en tiempo real. Muestra diferentes aspectos del sistema, como la memoria y el uso de CPU de procesos individuales. Para ver el estado del proceso, ejecutemos arriba en la terminal:

```
Tasks: 183 total,   1 running, 182 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.7 us,  1.1 sy,  0.0 ni, 97.1 id,  0.4 wa,  0.0 hi,  0.7 si,  0.0 st
MiB Mem :   3936.4 total,   1925.0 free,    850.6 used,   1160.8 buff/cache
MiB Swap:   2048.0 total,   2048.0 free,      0.0 used.   2834.2 avail Mem

    PID USER  PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   2237 bob   20   0  252252  81740  49204 S   2.3   2.0   0:09.37 Xorg
   2519 bob   20   0 3428664 375256 125080 S   2.0   9.3   0:19.57 gnome-shell
   2909 bob   20   0  966852  49944  37308 S   1.0   1.2   0:02.28 gnome-terminal-
      1 root  20   0  103500  13312   8620 S   0.7   0.3   0:04.44 systemd
   3588 bob   20   0   20600   3936   3380 R   0.3   0.1   0:00.01 top
      2 root  20   0       0      0      0 S   0.0   0.0   0:00.00 kthreadd
      3 root   0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
```

En la sección inferior de la salida del comando superior, podemos encontrar la columna S, que muestra el estado de cada proceso. A diferencia del comando ps, el comando top muestra el estado de cada proceso sin atributos de proceso adicionales.

El pseudoarchivo /proc

El pseudo sistema de archivos /proc contiene toda la información sobre los procesos en nuestro sistema. Por lo tanto, podríamos leer directamente el estado de un proceso a través de este pseudo sistema de archivos. La desventaja de este enfoque es que primero necesitaremos conocer el PID del proceso antes de poder leer su estado.

Para obtener el estado de un proceso, podemos extraer el valor de su archivo de pseudo estado en /proc/{pid}/status. Por ejemplo, podemos obtener el estado del proceso con PID 2519 leyendo el archivo /proc/2519/status:

```
$ cat /proc/2519/status | grep State
State:  S (sleeping)
```