

PROGRAMADOR DE TAREAS LINUX

Trabajo de Alejandro Sainz Sainz

SISTEMAS
INFORMÁTICOS
24-25

Alejandro Sainz Sainz

INTRODUCCIÓN	3
BREVE COMIENZO	4
COMIENZO DE LA TAREA	4
PROGRAMANDO TRES TAREAS NORMALES	5
TAREA 1	5
TAREA 2	6
TAREA 3	8
PROGRAMANDO TAREAS PERIODICAS	8
CONCLUSIÓN Y REFLEXIÓN	9

TABLA DE FIGURAS

Ilustración 1 Crontab -l primera vez.....4

Ilustración 2 Crontab -e primera vez.....4

Ilustración 3 nano5

Ilustración 4 Tarea 1.....5

Ilustración 5 Crontab -l segunda vez6

Ilustración 6 Tarea 2.....6

Ilustración 7 Creación de archivo_1.....7

Ilustración 8 Ejecución de la tarea programada.....7

Ilustración 9 Tercer Comando8

Ilustración 10 Tarea P1 - copia de seguridad.....8

Ilustración 11 Tarea P2 - Update y upgrade.....8

Ilustración 12 Tarea P3 - reboot del sistema9

INTRODUCCIÓN

Para hacernos una idea de lo que se va a mostrar aquí, tendremos que ponernos en la mente de un técnico de sistemas operativos, de sus acciones y de los mantenimientos requeridos por estos sistemas.

La programación de tareas en sistemas operativos, ya sea Linux como en nuestro caso u otro sistema, es una parte vital para el mantenimiento de los mismos. Esto se debe a que para mantener el correcto funcionamiento de estos elementos cruciales se deben de realizar grandes cantidades de tareas repetitivas, largas y que requieren de poca o nula atención por parte de los usuarios. Para este fin, se crean estas tareas automatizadas, ya sea mediante scripts o programas de terceros, que nos permitirán dedicarnos a otras labores mientras estas tareas se realizan de forma desatendida y en la fecha y hora que nosotros hayamos programado.

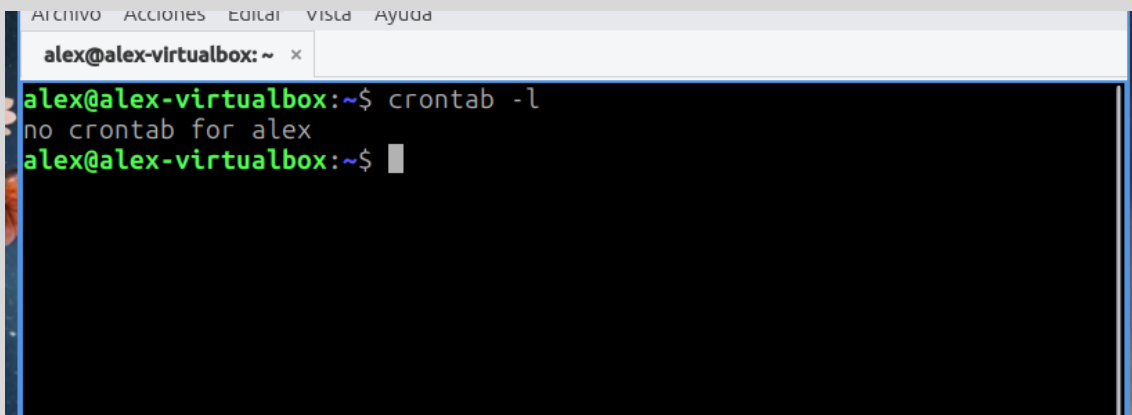
En nuestro caso vamos a ver un sistema de programación que nos proporciona el propio sistema, en el cual nosotros marcaremos los comandos para indicar el tipo de tareas y sus tiempos de ejecución, lo que nos da versatilidad, al no depender de las funcionalidades de aplicaciones de terceros, y seguridad al saber nosotros exactamente lo que se está ejecutando, ya que las aplicaciones externas pueden ejecutar código que nosotros desconocemos y no podemos controlar. Sin más que decir, procedemos a comenzar el ejercicio.

BREVE COMIENZO

Como ya se indica en el enunciado del ejercicio, no debemos de realizar los puntos 1 y 2. Son pequeñas explicaciones del sistema que vamos a usar, en este caso Cron o Crontab.

COMIENZO DE LA TAREA

Al principio, en tarea 1 y 2 se nos indica como abrir crontab y como introducir comandos.

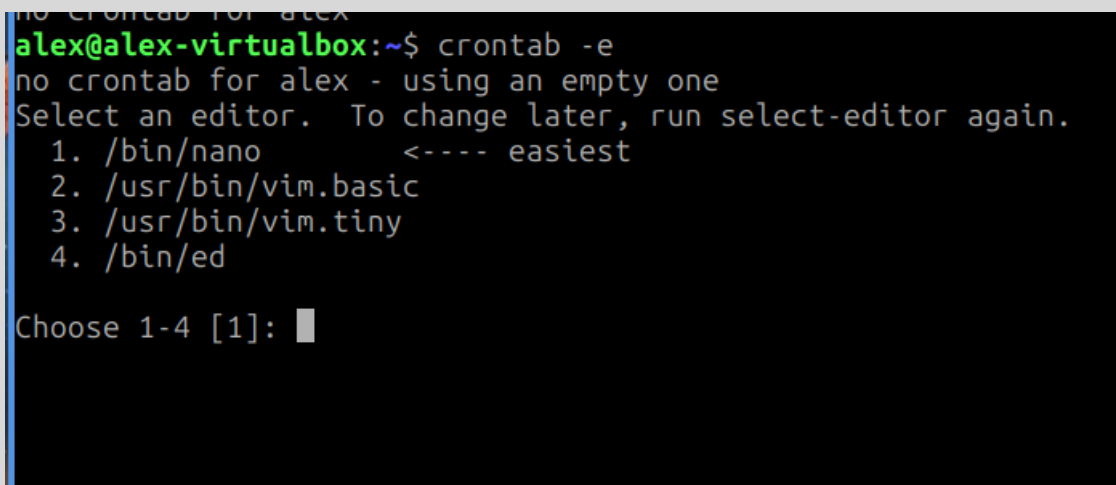


```
alex@alex-virtualbox: ~  
alex@alex-virtualbox:~$ crontab -l  
no crontab for alex  
alex@alex-virtualbox:~$
```

Ilustración 1 Crontab -l primera vez

Bien, la primera vez que ejecutas este comando, como no has generado ningún archivo crontab, me dice lo que se ve en la imagen. Que no tengo ningún archivo de este tipo disponible.

Acto seguido, en el enunciado nos indica que usemos el comando crontab -e. Esto abrirá ese archivo si nuestro usuario tiene creado 1, en caso contrario, creará uno nuevo.



```
alex@alex-virtualbox:~$ crontab -e  
no crontab for alex - using an empty one  
Select an editor. To change later, run select-editor again.  
1. /bin/nano          <---- easiest  
2. /usr/bin/vim.basic  
3. /usr/bin/vim.tiny  
4. /bin/ed  
Choose 1-4 [1]:
```

Ilustración 2 Crontab -e primera vez

Alejandro Sainz Sainz

La primera vez que ejecutamos este comando nos pide que elijamos el editor. En este caso yo he elegido nano, ya que es muy sencillito.

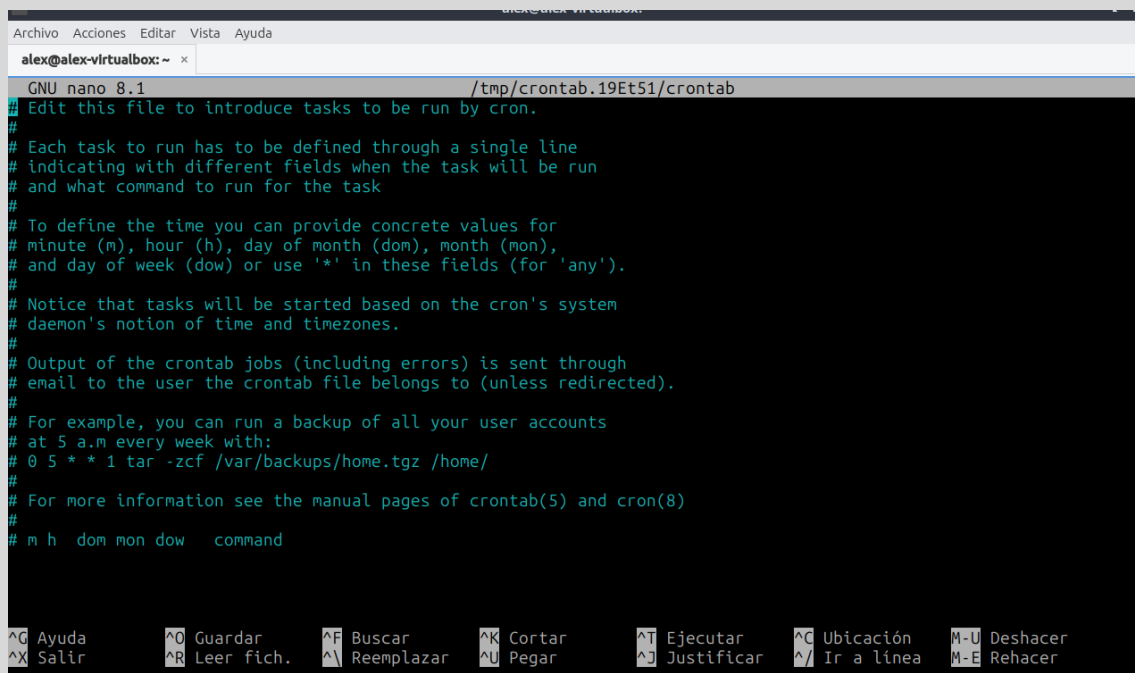


Ilustración 3 nano

Y esto es lo que se nos muestra por pantalla al elegir nano como editor. Una breve descripción en inglés de cómo funciona este archivo y de cómo insertar los temporizadores para el comando correspondiente.

PROGRAMANDO TRES TAREAS NORMALES

TAREA 1

Crear un archivo de texto cada día a las 9.

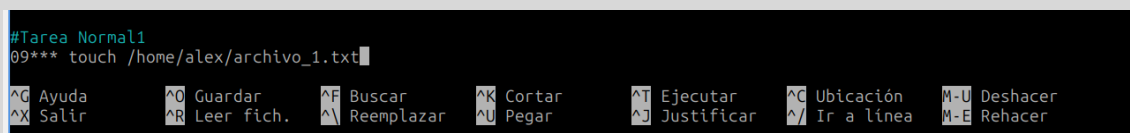
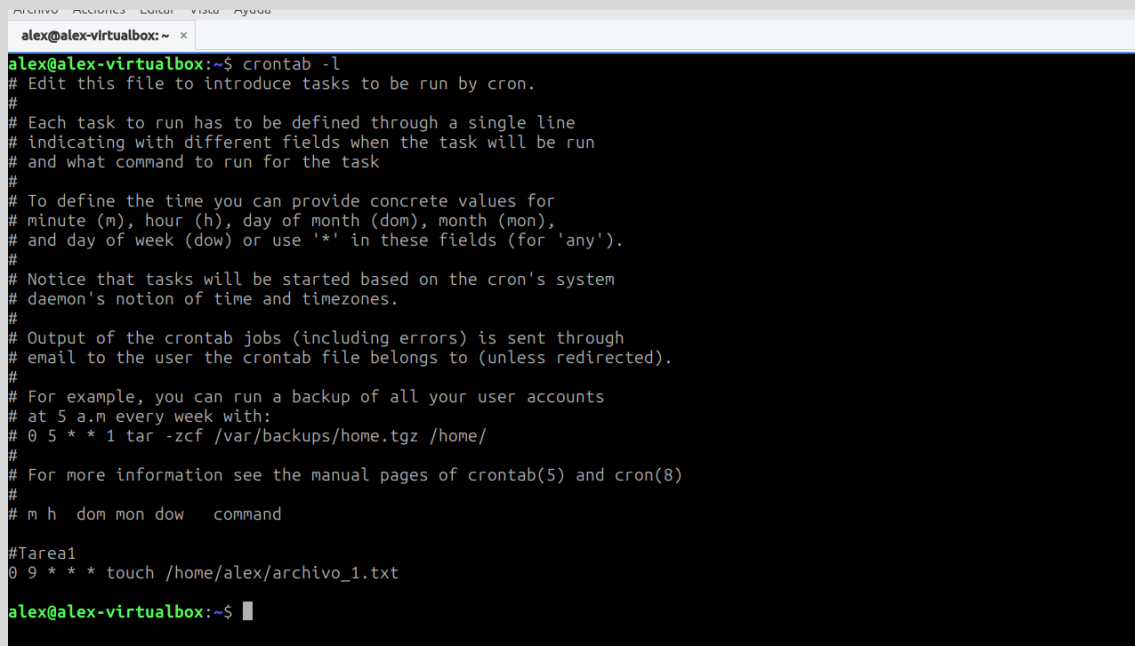


Ilustración 4 Tarea 1

Así es como se introduce el primer comando. Después, CTRL + O para guardar y CTRL + X para salir.

Como son las primeras veces, al salir, ejecuto el comando `crontab -l` para visualizar los ficheros que tengo creados para este usuario.

Alejandro Sainz Sainz



```
alex@alex-virtualbox: ~  
alex@alex-virtualbox:~$ crontab -l  
# Edit this file to introduce tasks to be run by cron.  
#  
# Each task to run has to be defined through a single line  
# indicating with different fields when the task will be run  
# and what command to run for the task  
#  
# To define the time you can provide concrete values for  
# minute (m), hour (h), day of month (dom), month (mon),  
# and day of week (dow) or use '*' in these fields (for 'any').  
#  
# Notice that tasks will be started based on the cron's system  
# daemon's notion of time and timezones.  
#  
# Output of the crontab jobs (including errors) is sent through  
# email to the user the crontab file belongs to (unless redirected).  
#  
# For example, you can run a backup of all your user accounts  
# at 5 a.m every week with:  
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/  
#  
# For more information see the manual pages of crontab(5) and cron(8)  
# m h dom mon dow   command  
#Tarea1  
0 9 * * * touch /home/alex/archivo_1.txt  
alex@alex-virtualbox:~$
```

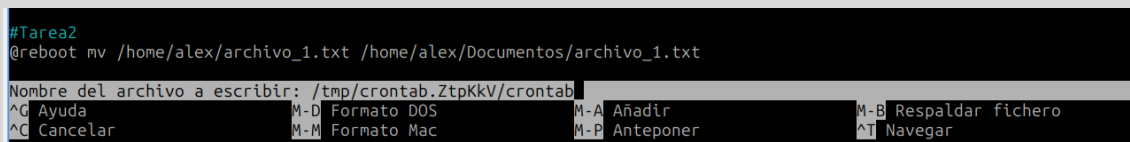
Ilustración 5 Crontab -l segunda vez

Cuando ejecuto este comando después de editar un archivo crontab, aparece esto en la terminal. Nos muestra todo el archivo que tenemos guardado.

TAREA 2

Mover un archivo cada vez que se reinicie el ordenador.

Seguimos el mismo procedimiento que en el ejercicio anterior. Volvemos a abrir nuestro archivo con crontab -e y al lío.

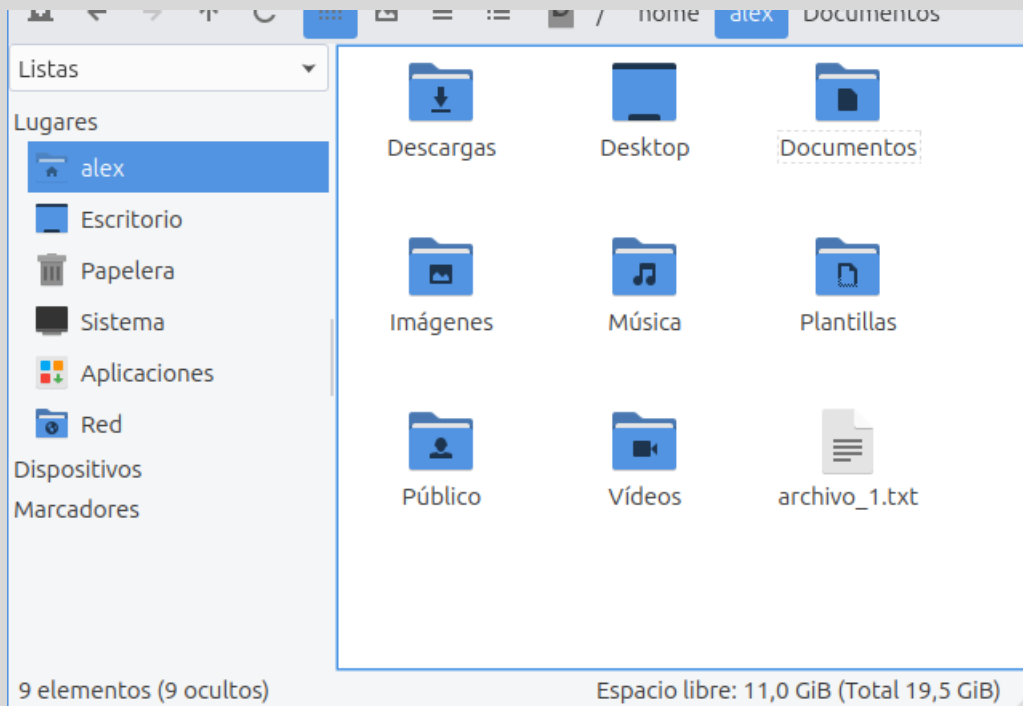


```
#Tarea2  
@reboot mv /home/alex/archivo_1.txt /home/alex/Documentos/archivo_1.txt  
Nombre del archivo a escribir: /tmp/crontab.ZtpKkv/crontab  
^G Ayuda      M-D Formato DOS  M-A Añadir     M-B Respalda fichero  
^C Cancelar   M-M Formato Mac  M-P Anteponer  ^I Navegar
```

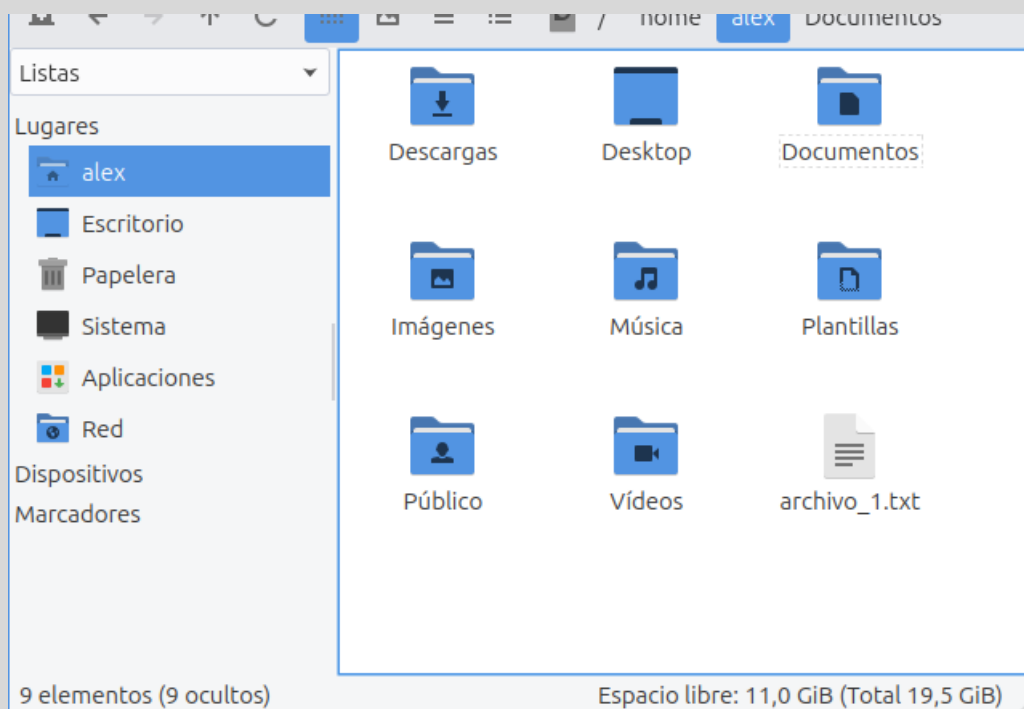
Ilustración 6 Tarea 2

Bien, en la imagen se ve como lo he hecho.

Ahora bien. En principio ninguna de estas tareas se puede comprobar muy bien. Salvo esta, que lo que podemos hacer es crear nosotros el archivo y reiniciar para ver como se ha movido de sitio.

**Ilustración 7 Creación de archivo_1**

Pues así, a lo bruto, creo el archivo que necesitamos. Después de eso, procedo a reiniciar.

**Ilustración 8 Ejecución de la tarea programada**

Al reiniciar, si voy a la carpeta en la que indiqué que quería que se moviese el archivo, veo que está ahí situado. Por lo tanto, estas cosas funcionan, siempre y cuando sepas lo que quieres hacer con el comando.

TAREA 3

Como ya he dicho esto, no se puede comprobar ninguno de los ejemplos. Así que lo único que voy haciendo es insertar los comandos, y comprobando con `crontab -l` si se ha guardado bien.

```
#Tarea3
0 18 * * 5 echo "Bienvenido al fin de semana" | wall
```

Ilustración 9 Tercer Comando

Con este comando ya tenemos una forma de que alguien nos de la bienvenida al fin de semana.

PROGRAMANDO TAREAS PERIODICAS

En este momento, para ahorrar un poco de tiempo, como los siguientes pasos son todos iguales, simplemente voy a poner las capturas de las tareas.

```
#Tarea Periodica 1
0 * * * * rsync -av /home/alex/Descargas /home/alex/Documents
```

Ilustración 10 Tarea P1 - copia de seguridad

```
#Tarea Periodica 2
0 3 * * * apt update && apt upgrade -y
```

Ilustración 11 Tarea P2 - Update y upgrade

```
#Tarea Periodica 3  
0 2 * * 0 reboot
```

Ilustración 12 Tarea P3 - reboot del sistema

Como podemos ver en los ejemplos anteriores, la variedad de tareas nos muestra la versatilidad de la herramienta. Si bien es cierto, como dije antes, que el mayor problema es no poder comprobar prácticamente ninguna de ellas, salvo que cambiemos ciertos parámetros para que ocurra todo a los pocos minutos. Aunque en un MV, por ejemplo, en nuestro caso, si programamos una actualización al poco de crear el comando, al ser un SO recién instalado, podemos tirarnos toda la mañana y no avanzaríamos nada. Pero bueno, es lo que tienen estas cosas.

CONCLUSIÓN Y REFLEXIÓN

Bien, pequeños comentarios. Lo intenté hacer de primeras en el sistema operativo que yo tengo instalado, pero me di cuante de que algunos comandos van distintos y la cosas no se iba a ver bien, sobre todo en el caso del update y el upgrade, ya que en Open Suse se usa Yast en vez de Apt.

Lo que me gusta de este tipo de herramientas, sobre todo en Linux, es la sencillez. No necesitas grandes entornos gráficos, aunque si es verdad que estos ayudan mucho. Pero ver como con un simple editor de texto ya podemos programar tareas rutinarias, al igual que en CMD con los .bat, te muestra lo bien pensadas que están estas cosas.

Se han diseñado estos sistemas para ejecutar una tarea concreta sin pensar mucho en el diseño, aunque quizá para tareas más complejas se queden un poco cortos, eso no lo sé a ciencia cierta.

Si bien es cierto que para sacarles el máximo rendimiento debes de conocer gran cantidad de comandos y el cómo combinar ciertas tareas para crear ciertos efectos, con unos pocos comandos ya podemos tener preparada la estructura base del funcionamiento de nuestro sistema.