

Veamos como sacar el máximo partido al histórico de comandos de linux

- Mostrar el timestamp del comando: normalmente cuando ejecutas history ves el número del comando y el comando propiamente dicho. Para propósitos de auditoría también te puede ser útil ver fecha y hora de cuando se ejecutó el comando. Para hacer esto, hay que setear la variable de entorno HISTTIMEFORMAT.

```
# export HISTTIMEFORMAT='%F %T '
# history | more
1 2008-08-19 20:01:01 service network restart
2 2008-08-19 20:02:09 exit
3 2008-08-19 20:08:39 id
4 2008-08-19 20:11:23 cat /etc/inetd.conf
```

- Búsqueda en el history: presionando Ctrl+R puedes buscar en tu history por un comando. Una vez presionado Ctrl+R en el prompt, empiezas a copiar el comando y el shell buscará por un comando que coincida en parte con lo que estás ingresando. Una vez que de con el comando que buscas puedes presionar Enter para confirmarlo o las flechas izquierda o derecha para editar el comando antes de ejecutarlo.

```
# (reverse-i-search)`inet': cat /etc/inetd.conf
```

- Repetir el último comando: puedes repetir el último comando de cuatro formas diferentes: presionando la flecha hacia arriba y Enter, ingresando el operador !! (bang bang) y Enter, ingresando !-1 y Enter o presionando Ctrl+P y Enter.
- Ejecutar un comando específico del history: para ejecutar un comando específico del history, debes conocer el número del mismo y luego anteceder al número del mismo el operador !

```
# history | more 1 service network restart 2 exit 3 cat /etc/services
```

```
# !3 cat /etc/services
```

- Ejecutar un comando previo que empieza con una palabra específica: ingresando ! seguido de unas cuantas letras, ejecutarás nuevamente un comando, que empiece con esas letras, previamente ejecutado.

```
# !ps ps ax | grep cron 5023 ? Ss 0:00 /usr/sbin/cron 8057 pts/0 R+ 0:00 grep cron
```

- Controlar la cantidad total de líneas en el history: seteando la variable de entorno HISTSIZE, en el .bash\_profile por ejemplo, podrás limitar la cantidad de líneas almacenadas por el history.

```
# nano ~/.bash_profile HISTSIZE=450 HISTFILESIZE=450
```

- Cambiar el nombre del archivo history: por default, el history del bash se almacena en el archivo ~/.bash\_history, para cambiar el nombre del mismo, debemos setear la variable de entorno HISTFILE:

```
# nano ~/.bash_profile HISTFILE=$HOME/.historia
```

- Eliminar las entradas contiguas repetidas: seteando la variable de entorno HISTCONTROL se pueden eliminar las entradas contiguas repetidas, mira el ejemplo:

```
# ls # ls # ls # history | tail -4 353 ls 354 ls 355 ls 356 history | tail -4 # # export
```

```
HISTCONTROL=ignoredups # ls # ls # ls # history | tail -3 357 export HISTCONTROL=ignoredups 358  
ls 359 history | tail -3
```

- Borrar duplicados a lo largo del history: el valor ignoredups de la variable de entorno HISTCONTROL utilizado anteriormente solo borra entradas duplicadas consecutivas, para eliminar entradas duplicadas en todo el archivo hay que setear HISTCONTROL=erasedups
- Forzar a que el history no recuerde un comando en particular: seteando la variable HISTCONTROL con el valor ignorespace, puedes hacer que el history no recuerde un comando en particular si al mismo le dejas un espacio en blanco delante.

```
# export HISTCONTROL=ignorespace # ls -l # pwd # service httpd stop # history | tail -3 567 ls -ltr  
568 pwd 569 history | tail -3
```

- Limpiar todas las entradas previas del history: ejecutando history -c eliminarás para la consola activa todas las entradas previas al history, pero no estarás eliminándolas del bash\_history, por lo tanto en una nueva consola conservarás el history.
- Sustituir palabras de los comandos del history: a veces puede ser que necesites ejecutar otro comando, pero con el mismo argumento que el comando anterior. En el siguiente ejemplo, el !:\$ a continuación del comando nano obtiene el argumento del comando anterior:

```
# ls .bash_logout .bash_logout # vi !:$ vi .bash_logout
```

En este otro ejemplo, el !^ a continuación del comando nano obtiene el primer argumento del comando anterior:

```
# cp .bash_logout .bash_logout.old # vi !^ vi .bash_logout
```

- Sustituir un argumento específico de un comando específico: en el siguiente ejemplo, !cp:2 busca por el comando anterior que empieza con cp y toma el segundo argumento y lo sustituye para el comando cat que se ejecuta a continuación:

```
# cp .bash_logout .bash_logout.old # cat !cp:2 cat .bash_logout.old
```

- Deshabilitar el uso del history: si quieres deshabilitar el history y que el bash shell no recuerde más los comandos ingresados, debes setear la variable HISTSIZE en 0

```
# export HISTSIZE=0 # history # #nota aquí que no se muestra nada
```

- Ignorar comandos específicos: muchas veces puedes no querer agregar al history comandos usuales como pwd, ls, etc. Para ignorar comandos debemos setear la variable HISTIGNORE (no sería mala idea hacerlo desde el ~/.bash\_profile) de la siguiente manera:

```
# export HISTIGNORE="pwd:ls:ls -ltr:"
```

## Referencias

<http://luauf.com/2008/08/19/utilizar-el-bash-history-a-fondo/>

From:

<https://intrusos.info/> - LCWIKI

Permanent link:

<https://intrusos.info/doku.php?id=linux:history&rev=1290086213>

Last update: **2023/01/18 13:55**

