

Algo de Linux

Debian, Ubuntu, Linux en general, software libre, el shell de linux, scripts bash...

- Página principal
- Github
- Posts privados
- MultCloud
- Google Keep
- Guerrillamail.com

jueves, 3 de abril de 2008

El shell de linux: Comando grep

El comando **grep** nos permite buscar, dentro de los archivos, las líneas que concuerdan con un patrón. Bueno, si no especificamos ningún nombre de archivo, tomará la entrada estándar, con lo que podemos encadenarlo con otros filtros.

Por defecto, grep imprime las líneas encontradas en la salida estándar. Es decir, que podemos verlo directamente la pantalla, o redireccionar la salida estándar a un archivo.

Como tiene muchísimas opciones, vamos a ver tan sólo las más usadas:

- c** En lugar de imprimir las líneas que coinciden, muestra el número de líneas que coinciden.
- e** PATRON nos permite especificar varios patrones de búsqueda o proteger aquellos patrones de búsqueda que comienzan con el signo **-**.
- r** busca recursivamente dentro de todos los subdirectorios del directorio actual.
- v** nos muestra las líneas que no coinciden con el patrón buscado.
- i** ignora la distinción entre mayúsculas y minúsculas.
- n** Numera las líneas en la salida.
- E** nos permite usar expresiones regulares. Equivalente a usar **egrep**.
- o** le indica a grep que nos muestre sólo la parte de la línea que coincide con el patrón.
- f** ARCHIVO extrae los patrones del archivo que especifiquemos. Los patrones del archivo deben ir uno por línea.
- H** nos imprime el nombre del archivo con cada coincidencia.

Veamos algunos ejemplos:

- Buscar todas las líneas que contengan palabras que comiencen por **a** en un archivo:
`$ grep "\<a.*\>" archivo`

Otra forma de buscar, sería:
`$ cat archivo | grep "\<a.*\>"`

- Mostrar por pantalla, las líneas que contienen comentarios en el archivo `/boot/grub/menu.lst`:
`$ grep "#"/boot/grub/menu.lst`

- Enviar a un fichero las líneas del archivo `/boot/grub/menu.lst` que no son comentarios:
`$ grep -v "#"/boot/grub/menu.lst`

- Contar el número de interfaces de red que tenemos definidos en el fichero `/etc/network/interfaces`:
`$ grep -c "iface" /etc/network/interfaces`

- Mostrar las líneas de un fichero que contienen la palabra **BADAJOS** o **HUELVA**:
`$ grep -e "BADAJOS" -e "HUELVA" archivo`

- Mostrar las líneas de un fichero que contienen la palabra **BADAJOS** o **HUELVA**, numerando las líneas de salida:
`$ grep -n -e "BADAJOS" -e "HUELVA" archivo`

- Mostrar los ficheros que contienen la palabra **TOLEDO** en el directorio actual y todos sus subdirectorios:
`$ grep -r "TOLEDO" *`

Veamos algunos ejemplos con expresiones regulares:

- Obtener la dirección MAC de la interfaz de red `eth0` de nuestra máquina:
`$ ifconfig eth0 | grep -oiE '([0-9A-F]{2}){5}[0-9A-F]{2}'`

Sacamos la dirección MAC de la interfaz `eth0` de nuestra máquina haciendo un:
`ifconfig eth0`

Y aplicando el filtro grep:
`grep -oiE '([0-9A-F]{2}){5}[0-9A-F]{2}'`

Buscar en este blog

Buscar

Páginas vistas en total



3,234,540

Tu programa

Cómo lo ven los usuarios

Interfaz

Lógica

Lenguaje

Interfaz

MAGIA

W

Wikingenería

@Wikingenería

15:39 - 31 ene. 2017

6

376

472

Blogs de compañeros administradores

- [Blog de Alfonso Pastor](#)
- [Blog de José Miguel Medina](#)
- [Blog de Manuel Gómez](#)

Traducir

Seleccionar idioma

▼

Con la tecnología de [Google Traductor de Google](#)

Sígueme en las redes sociales

Me gusta

A 341 personas les gusta esto. Sé el primero de tus amigos.

Seguir a @algotelinux

203 seguidores

Esteban M. Navas ...

Seguir

Algo de Linux directamente a tu e-mail

ingresa tu e-mail

Suscríbete



Seguidores

-E Indica que vamos a usar una expresión regular extendida.

En cuanto a la expresión regular, podemos dividirla en dos partes:
[0-9A-F]{2:}{5} Buscamos 5 conjuntos de 2 caracteres seguidos de dos puntos
[0-9A-F]{2} seguido por un conjunto de dos caracteres.

Como las direcciones MAC se representan en hexadecimal, los caracteres que buscamos son los números del 0 al 9 y las letras desde la A a la F.

- Extraer la lista de direcciones de correo electrónico de un archivo:
grep -Eio '[a-z0-9_-]*@[a-z0-9_-]*[a-z]{2,4}' fichero.txt

Utilizo las mismas opciones que en el caso anterior:

-o Indica que la salida del comando debe contener sólo el texto que coincide con el patrón, en lugar de toda la línea, como es lo habitual.
-i Lo he usado para que ignore la distinción entre mayúsculas y minúsculas.
-E Indica que vamos a usar una expresión regular extendida.

Analicemos ahora la expresión regular:
[a-z0-9_-]*@[a-z0-9_-]*[a-z]{2,4}

Al igual que antes, la vamos dividiendo en partes:
[a-z0-9_-]* Una combinación de letras, números, y/o los símbolos . _ y - de uno o más caracteres
@ seguido de una arroba
[a-z0-9_-]* seguido de una cadena de letras, números y/o los símbolos . y -
[a-z]{2,4} seguido de una cadena de entre dos y cuatro caracteres.

- Obtener la dirección IP de la interfaz de red eth1 de nuestra máquina:
\$ ifconfig eth1 | grep -oIE '([0-9]{1,3}\.){3}[0-9]{1,3}' | grep -v 255

En el ejemplo anterior, hemos tomado la información que nos ofrece ifconfig:
ifconfig eth1

Hemos filtrado dicha información con el comando grep, obteniendo todas las direcciones IP que aparecen:
grep -oIE '([0-9]{1,3}\.){3}[0-9]{1,3}'

Por último, hemos filtrado la salida del comando anterior, para eliminar la dirección de broadcast junto con la máscara de red para quedarnos sólo con la dirección IP de la máquina:
grep -v 255

La línea anterior no mostraría las líneas que no contengan el valor 255, es decir, las direcciones de broadcast y máscara de red.

Analicemos ahora el comando grep:
grep -oIE '([0-9]{1,3}\.){3}[0-9]{1,3}'

Al igual que en los otros dos ejemplos de expresiones regulares uso las opciones -oIE en el comando grep:

-o Indica que la salida del comando debe contener sólo el texto que coincide con el patrón, en lugar de toda la línea, como es lo habitual.
-i Lo he usado para que ignore la distinción entre mayúsculas y minúsculas.
-E Indica que vamos a usar una expresión regular extendida.


En cuanto a la expresión regular:
'([0-9]{1,3}\.){3}[0-9]{1,3}'
([0-9]{1,3}\.){3} Representa 3 bloques de entre uno y tres dígitos separados por puntos. Observemos que como el punto es un metacaracter, tengo que usar el caracter de escape \ para que no sea interpretado como un metacaracter, sino como un caracter normal.
[0-9]{1,3} Representa el último bloque de la dirección IP, que está formado por un número de entre 1 y 3 dígitos.


Publicado por Esteban M. Navas Martín en 18:55



Etiquetas: comandos, linux, scripts

24 comentarios:

 **Francisco** dijo...
exlente me sirvió de mucho gracias :D
27 de diciembre de 2010, 3:35

 **Miguel Ortiz** dijo...
Están bien las explicaciones pero el uso de egrep y fgrep es obsoleto.
Saludos.
1 de junio de 2011, 14:40



Seguir

Etiquetas

adb (1) alternativas (3) android (16) apt (9) apt-mirror (2) autofs (3) backtrack (2) backup (13) bdd (1) bind9 (1) bios (10) bittorrent (3) bluetooth (1) bootmanager (6) bots (3) bugs (5) certificados (1) cheatsheets (2) chocolatey (11) cinnamon (2) clonacion (29) cloud (2) cluster (1) clusterssh (2) comando (1) comandos (187) correo (1) cups (15) cursos (1) d5100 (1) debian (4) debootstrap (1) denyhosts (4) distros (17) django (3) dlink (1) drbl (1) eeePC (2) efi (7) elearning (6) eoptes (2) errores (42) errores (2) extremadura (4) facter (2) filesystems (3) find (1) firefox (28) firmware (37) fonts (1) freebsd (1) freedos (4) freeradius (1) gam (1) gdm3 (1) gimp (1) git (1) gnome (28) gnome-shell (6) google-chrome (9) gpg (3) gpo (6) grub (14) hardware (56) herramientas (121) hp5730 (2) hpprodesk (4) hpx360 (3) i915 (2) iceweasel (5) ide (1) ies (144) impresoras (5) infolab (3) jessie (18) kernel (3) keyring (2) kvm (2) latex (1) ldap (17) libreoffice (8) libros (3) lightdm (1) linux (338) locales (1) lowi (1) ltsp (5) lvm (1) lyx (1) mega (1) mirrors (8) modulos (6) monit (3) moviles (5) msi (1) mtk (2) multiarch (1) mysql (1) nautilus (5) nds (1) nmap (3) nook (2) ntp (4) ocsinventory (1) openmediavault (7) openwrt (20) opera (1) p610 (4) pam (2) pantallainformativa (2) paquetes (39) pdc (16) pfsense (12) pkgsync (30) plymouth (1) portables (1) postfix (1) powerdns (3) powershell (5) problemas (1) procedimientos (2) proxmox (13) pstools (1) pulseaudio (1) puppet (106) python (7) raspberry pi (12) recursos (46) redes (45) refind (10) repositorios (12) root (2) rsat (2) s5570i (1) samba (9) scripts (113) sed (1) seguridad (22) serviciosweb (5) servidores (45) shell (39) siatic (2) smartboard (1) software (231) squeeze (15) squid (8) ssh (6) smtp (1) switches (2) syslinux (1) telegram (3) testdisk (1) trusty (7) tti (1) ubiquiti (7) ubuntu (34) upstart (2) vexia (4) virtualbox (1) virtualizacion (18) wheezy (19) wifi (39) wifislax (3) wifiway (2) wii (4) windows (96) wine (1) woxtertv300 (2) wsus (2) xenial (2) xfce (5) xiaomi (1) xml (1) xorg (5) xrandr (1) xubuntu (8) yumi (1) zentyal (7)

Así mismo fgrep también está en desuso y en su lugar debe utilizarse grep -F.

Pero eso no quiere decir que no puedas usar las opciones equivalentes con el comando grep que sigue estando en uso y todas sus opciones son muy útiles.

1 de junio de 2011, 16:05



Miguel Ortiz dijo...

Esteban, hacía referencia a:

("-E nos permite usar expresiones regulares. Equivalente a usar egrep.")

No dije que no se pudiera usar grep -E, más bien aclarar que ya no se usa egrep.

Saludos.

1 de junio de 2011, 16:16



Esteban M. Navas dijo...

Ok. No había entendido lo que querías decir.

1 de junio de 2011, 20:58



rb34 dijo...

Gracias, justo lo que buscaba...

27 de junio de 2011, 4:59

Anónimo dijo...

muy útil asias

5 de agosto de 2011, 14:46



Aaron dijo...

Eso ni es importante... Lo importante es que si uno busca "a*" no son las palabras que comiencen con a, son las palabras que CONTENGAN a, Tremendo error!! y en el primer ejemplo!!

6 de agosto de 2011, 0:53

Antonio Castro dijo...

Buenas tardes, quería hacerle una consulta. Esto es si se ejecuta desde el Terminal de Linux pero en el caso que yo lo quiera hacer desde un script .sh ¿cómo sería la llamada al ifconfig y el uso del grep?

6 de agosto de 2011, 22:57



Esteban M. Navas dijo...

Hola, Aaron. No es necesario que escribas en mayúsculas. Como ya sabes, eso significa gritar. Tampoco es tan tremendo el error... Este blog era tan sólo para mí, pero decidí dejarlo público, por si a alguien más le era útil. Si no te gusta, nadie te obliga a consultarlo.

A veces cuando he escrito algo en el blog ha sido a partir de notas en papel de algo que había probado y después de mucho tiempo, con lo que se me ha colado alguna errata.

En cualquier caso, te agradezco tu aportación. Queda corregida la errata para que el ejemplo refleje la búsqueda de palabras que comienzan por "a". Cuando escribí el artículo, me equivoqué a la hora de redactarlo. Lo que tenía escrito en papel era buscar las líneas que comiencen por "a", claro que también se me pasó escribir el símbolo de comienzo al transcribir desde papel.

8 de agosto de 2011, 10:02



Esteban M. Navas dijo...

Hola, Antonio. Los comando que ejecutas en un terminal puedes usarlos también dentro de un script, de la misma forma.

8 de agosto de 2011, 10:05



Ariel dijo...

Hola, Muchachos, estoy queriendo hacer un scrip en bash para manejar un texto.txt estoy usando el sed pero este me corta líneas, necesito si hay algun comando para que me corte el texto. por ejemplo en el primer punto (.) gracias

19 de octubre de 2011, 14:50



Esteban M. Navas dijo...

No sé qué es lo que necesitas exactamente, pero supongo que te refieres al comando cut:

<http://enavas.blogspot.com/2008/02/el-shell-de-linux-comando-cut.html>

Por ejemplo:

REFINO: Cambiar el icono de un bootloader dlvr.it/Pvv9fC

Algo de Linux

Aprende a administ...
enavas.blogspot.com

39min

Esteban M. Navas retweeteó



Albert Rivera

@Albert_Rivera

Ningún padre decente puede tolerar que se adoctrine y señale a niños en la escuela. Todo mi apoyo a las familias acosadas.

No estáis solos.

14 oct. 2017

Esteban M. Navas retweeteó



ESLE

@esletweets

Insertar

Ver en Twitter

Archivo del blog

- ▶ 2017 (193)
- ▶ 2016 (236)
- ▶ 2015 (126)
- ▶ 2014 (165)
- ▶ 2013 (147)
- ▶ 2012 (114)
- ▶ 2011 (26)
- ▶ 2010 (37)
- ▶ 2009 (26)
- ▼ 2008 (137)
 - ▶ diciembre (9)
 - ▶ noviembre (16)
 - ▶ octubre (31)
 - ▶ septiembre (17)
 - ▶ agosto (4)
 - ▶ julio (1)
 - ▶ junio (5)
 - ▶ mayo (8)
 - ▼ abril (7)
 - XAMPP: Montar tu propio servidor web
 - Creación de actividades educativas con JClic
 - g4u: Clonar HDD en Linux
 - Wink: Una herramienta para crear tutoriales de for...
 - Autologin en linux con gnome
 - El shell de linux: Comando stat
 - El shell de linux: Comando grep
- ▶ marzo (15)
- ▶ febrero (15)
- ▶ enero (9)
- ▶ 2007 (47)

Anonimo dijo...

Se puede guardar el contenido mostrado por el comando GREP en una variable para ser usado despues?

25 de octubre de 2012, 13:08

**Esteban M. Navas dijo...**

Sí. Puedes guardar el resultado de ejecutar un grep en una variable.

26 de octubre de 2012, 17:45

Albano dijo...

Que bueno, Muy bien explicado, muchas gracias.

10 de noviembre de 2012, 18:01

Anónimo dijo...

Excelente, tras mucho buscar aquí he encontrado muy bien explicado el comando con la opción -E, me quito el sombrero.
Muchas gracias.

17 de enero de 2013, 13:43

Milton Padilla dijo...

Gracias por tan excelente post me ha sido de mucha ayuda.

11 de junio de 2013, 21:36

Anónimo dijo...

hola. muy bueno. una pregunta
y si es un grep recursivo en lugar de un archivo a una carpeta con subcarpetas y archivos, y todo lo que hay dentro... cómo sería
ejemplo:

Extraer la lista de direcciones de correo electrónico de un archivo:

```
grep -Eior '[a-z0-9_-]+@[a-z0-9_-]+[a-z]{2,4}' carpeta (con muchas subcarpetas y archivos)
```

o

cd carpeta

```
egrep -oir '[a-z0-9_-]+@[a-z0-9_-]+[a-z]{2,4}' *
```

o cómo???

gracias

5 de noviembre de 2015, 1:52

Barquisimeto dijo...

"grep -v", de lo mas útil que pueda haber a la hora de revisar un archivo de configuración.

12 de diciembre de 2015, 5:06

Andrea R dijo...

Necesito hacer un grep asi

ls -lrt | grep "Apr 28 " pero que el valor de busqueda este contenido en una variable , quedaria de esta manera ls -lrt | grep \$fecha el problema es que genera el siguiente error:

```
ls -lrt | grep $fecha3
```

```
grep: 0652-033 Cannot open 28.
```

```
grep: 0652-033 Cannot open ".
```

Solo funciona si el valor de la variable no tiene espacios :(, este es el valor de la variable, ya lo probe sin comillas y tampoco sirve

```
echo $fecha3
```

```
"Apr 28 "
```

Agradezco su ayuda si alguien sabe como lo puedo hacer, gracias.

29 de abril de 2016, 1:50

**Esteban M. Navas Martín dijo...**

Hola, Andrea. Lo que tienes que hacer es poner la variable entre comillas:

```
ls -lrt | grep "$fecha3"
```

29 de abril de 2016, 18:14

**Times dijo...**

¿Como puedo usar grep para que al listar el contenido de /proc me aparezcan todas las que no terminan en numero?
Muchas gracias

3 de diciembre de 2016, 16:25

Esteban M. Navas Martín dijo...

Enlaces a esta entrada

[Crear un enlace](#)

[Entrada más reciente](#)

[Página principal](#)

[Entrada antigua](#)

Suscribirse a: [Enviar comentarios \(Atom\)](#)

Tema Sencillo. Imágenes del tema: [mattjeacock](#). Con la tecnología de [Blogger](#).