



# Análisis de ficheros log en GNU/Linux

José Alberto Suárez  
suarez@wprensa.com

**E**ste taller nos ayudará a analizar y entender mejor los archivos log de nuestro sistema, desde el nivel más básico hasta aspectos un poco más avanzados. Vamos a ver tanto herramientas y procedimientos genéricos, como algunos específicos para logs concretos.

Se presuponen algunas nociones básicas, pero aun así vamos a dedicar la primera parte a describir de una forma generalizada cómo sacar partido de los, en ocasiones (por no decir siempre), largos y tediosos logs. Veremos algunos comandos, scripts y expresiones tanto de uso general, que podremos reutilizar en diversos escenarios, como para propósitos específicos.

En la segunda parte explicaremos cómo analizar unos logs concretos, así como diversas utilidades que nos pueden facilitar y mostrar de forma más gráfica la información que contienen éstos.

## LOS FICHEROS LOG

La necesidad de analizar los logs puede darse por varios motivos, a veces es solo para obtener información (¿¿, por simple curiosidad, etc...), pero principalmente se requieren para:

- ▶ La creación de gráficas y estadísticas de estado y uso (por ejemplo en servidores web, o cualquier otro tipo de servicio), habitualmente generadas por herramientas como Awestat o Webalizer.
- ▶ El análisis de errores y fallos, por ejemplo, para intentar averiguar por qué cierta aplicación no funciona como debiera.
- ▶ Por motivos de seguridad y prevención, caso del análisis de los logs de cortafuegos y otras herramientas de seguridad.
- ▶ Para automatizar tareas basadas en eventos como recibir un correo cuando se detecta que una IP concreta intenta hacer login mediante SSH.

## DEMONIOS

Los logs pueden consistir de distintas formas en el sistema, las más comunes son ficheros de texto plano ubicados en /var/log

Estos ficheros generalmente son creados y escritos por el demonio del sistema encargado de gestionar los logs. Hay una gran variedad, pero los más comunes son:

- ▶ **Syslog (syslogd)**: muy popular, suele estar instalado por defecto en muchas distribuciones.
- ▶ **Syslog-ng**: es la siguiente generación y prácticamente es igual a su antecesor, con la gran ventaja de que permite enviar y recibir logs desde y a otros sistemas, o guardar la información en bases de datos. Cada vez más distribuciones lo usan por defecto.
- ▶ **Metalog**: muy útil para sistemas portátiles, ya que no escribe continuamente al disco duro y así permite usar sistemas de ahorro de energía. Es muy sencillo de utilizar y configurar.

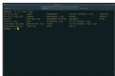
También es usual que los propios servicios que necesitan crear logs se encarguen de ello sin usar el demonio del sistema, normalmente es algo opcional. Cuando esto ocurre, dichos logs se configuran en el propio servicio, por ejemplo, Apache. Éste creará su propio directorio y gestionará sus propios logs si no le decimos lo contrario.

Los ficheros log también pueden encontrarse en forma de bases de datos binarias, por lo que requieren un programa externo para visualizarlos. En otras ocasiones, son directamente escritos en un servidor de bases de datos como MySQL, esto suele ser más común en redes un poco más complejas que necesitan centralizar los logs y visualizarlos conjuntamente.

## LOGROTATE

Aunque no es una herramienta de análisis, logrotate es una utilidad fundamental si no queremos que nuestro sistema se inunde literalmente de información. Esta aplicación hace que los logs "rotan", es decir, se copian y comprimen cada cierto tiempo, predetermi-

do para cada caso. Si no se hiciera así, veríamos como nuestro disco duro va menguando poco a poco hasta acabar con la partición donde se encuentren los logs. Generalmente logrotate guarda los logs siguiendo la forma nombredebg.N.gz, donde N es el número de veces que se ha rotado el log hasta un número máximo que normalmente es cinco. El número de rotaciones suele ser de una vez por semana o, para aquellos que crecen mucho, una vez por día.



Aspecto de logs rotados con logrotate.

## UN CASO ESPECIAL: DMESG

Como veníamos comentando, los logs generalmente son creados por el demonio del sistema encargado de ello. Pero esto nos plantea un problema, ¿qué ocurre con los logs que se generan antes de que dicho demonio sea iniciado, por ejemplo, en el arranque del sistema. De éstos se encarga



Salida típica del comando dmesg.

Los logs pueden ser generados por el demonio del sistema o por una aplicación y generalmente son largos y tediosos

dmseg y prácticamente contiene todo lo que ocurre en el arranque e inicio del sistema. Visualizar estos ficheros es tan simple como ejecutar el comando dmseg. Este nos dará como salida estándar todo el contenido, que por supuesto podemos tratar como cualquier otro log.

**MANOS A LA OBRA**

Para analizar los logs haremos uso de algunas herramientas que se pueden encontrar en cualquier distribución de Linux, y nos ayudaremos de tuberías y expresiones regulares. Los comandos que vamos a utilizar son:

- ▶ **Tuberías y redireccionadores:** |, >, <, >>, 2>, 2>&1,
- ▶ **less:** paginador al estilo more, pero con mejores resultados (otra opción es more).
- ▶ **cat:** lanza el contenido de un fichero a la salida estándar.
- ▶ **tail:** nos muestra las últimas diez líneas (o las que indiquemos con la opción -n).
- ▶ **head:** nos muestra las diez primeras líneas.
- ▶ **tr:** cambia un carácter por otro.
- ▶ **cut:** corta los caracteres indicados usando delimitadores.
- ▶ **sort:** ordena una lista y sus atribúticamente o numéricamente.
- ▶ **grep:** compara dos archivos.
- ▶ **comm:** filtra por expresiones regulares (y sus variantes [grep y egrep]).
- ▶ **awk:** es un poderoso lenguaje, basado en expresiones regulares, que nos permite filtrar, editar e imprimir.

**LESS**

El comando less es un paginador que nos muestra la información de un fichero por páginas, es decir, podemos leerla de manera más cómoda. El concepto es igual al de more, pero además less nos permite realizar búsquedas (f), abrir archivos comprimidos, y no sale hasta que no se lo indicamos (q). Otra aplicación similar a less es more, que está ganando terreno y tiene opciones muy interesantes. El uso de less es muy simple:

```
# less /var/log/fichero

Esto nos mostrará el contenido de ese fichero. Pero también podemos usarlo para filtrar la salida estándar:
```

```
# dmseg | less
```

**CAT**

Cat es uno de los comandos más usados, fue ideado para unir ficheros, y resulta muy versátil. Con cat podemos lanzar a la salida

estándar el contenido de cualquier fichero de Linux. Es muy útil para poder aplicar filtros a esa salida, o usar algún tipo de redireccionador:

```
# cat /var/log/fichero >> fichero2
```

**TAIL**

Sin duda, éste es el comando que más ejecutaremos a la hora de analizar un log. Un simple archivo puede tener cientos de líneas, ya que se pueden guardar varias horas o días en cada uno. Pero normalmente solo nos interesan los últimos eventos y para esto nada mejor que tail, ya que únicamente nos mostrará las últimas diez líneas. Su manejo es muy simple, podemos usarlo contra un archivo o recogiendo la salida estándar:

```
# tail /var/log/fichero

También podemos indicarle el número de líneas concretas que queremos que nos muestre:
```

```
# dmseg | tail -n10
```

Incluso tenemos la posibilidad de monitorizar en tiempo real un fichero, lo cual resulta tremendamente útil cuando estamos depurando fallos:

```
# tail -f /var/log/fichero
```

**HEAD**

Su funcionamiento es igual al de tail pero en lugar de mostrarnos las últimas líneas nos devuelve las primeras. También como el comando anterior, por defecto el número de líneas que muestra es diez:

```
# head -n10 /var/log/fichero
```

**TR**

El comando tr tiene normalmente un uso secundario, y nos resultará de utilidad a la hora de simplificar expresiones regulares. Básicamente lo usaremos para sustituir una cadena por otra de forma rígida. Sobre todo es útil cuando no tenemos ganas de ponernos a pensar una expresión regular adecuada o no nos acordamos de cómo utilizar sed. Su uso es bastante simple:

```
# cat fichero1
# hola,mundo
# cat fichero2 | tr "., " " "
# hola mundo
```

Ahora nos será más fácil pensar esa salida o filtrarla. Aunque esto normalmente también se puede conseguir afirmando un poco más los comandos grep o awk, en ocasiones nos puede ahorrar unos minutos.

**CUT**

El comando cut, como su nombre indica, nos permite "cortar" porciones de texto delimitadas por delimitadores. Como delimitador podemos establecer el carácter que nos sea más útil. Cada delimitador define un campo o columna que podemos cortar. Por defecto el comando cut usa como delimitador el contenido de la variable \$IFS (Input Field Separator). El uso no es complejo y puede resultar muy útil como comando secundario a la hora de filtrar logs:

```
# cat fichero
# 1,2,3,4
# cat fichero | cut -d " " -f2-4
# 1
```

En este caso hemos definido como delimitador la coma, y le hemos dicho que corte del campo dos al cuatro. También se puede usar para cortar solo caracteres:

```
# cat fichero | cut -d " " -c1-5,8,9
```

Esto cortaría los caracteres del 1 al 5 y el 8 y 9.

**SORT**

Este comando ordena las líneas que le demos, tanto dentro de un fichero como por la salida estándar. Nos puede ser de utilidad a la hora de ordenar una lista de IPs, de fechas, o un número de entradas al sistema. Tiene varias opciones interesantes como:

- ▶ **-c:** Ordena alfabéticamente. Ignora cualquier carácter que no sea una letra a la hora de ordenar.
- ▶ **-f:** Ignora la diferencia entre mayúsculas y minúsculas.
- ▶ **-n:** Ordena numéricamente.
- ▶ **-r:** Realiza un orden inverso.

En el siguiente ejemplo:

```
# cat fichero
# 2
# 3
# 1
# cat fichero | sort -nr
# 3
# 2
# 1
```

hemos ordenado numéricamente y en orden inverso.

**■ COMM**

Este comando es muy útil a la hora de comparar dos listas previamente ordenadas (con el comando `sort` por ejemplo). Imaginemos que hemos extraído a lo largo del día una serie de IPs del log del servidor, y queremos saber si coinciden con las IPs que extraíjamos la semana pasada. Al ejecutar el comando `comm` en estas dos listas nos devolverá tres columnas. La primera son las líneas únicas en el primer fichero, la siguiente, las líneas únicas en el segundo archivo, y la tercera las líneas comunes a ambos. Podemos suprimir cualquiera de esas columnas usando las opciones `-1`, `-2`, `-3`. Supongamos que solo queremos las líneas comunes:

```
# comm -1 -2 fichero1 fichero2
```

Usando el comando `sort` sin opciones, obtendremos unos ficheros adecuados para utilizar con la instrucción `comm`.

**■ GREP**

El comando `grep` nos ayudará a filtrar la misma información que contiene un log. Es muy flexible y funciona con expresiones regulares. Podemos añadir un filtro tanto como queramos, para obtener solo lo que nos interesa. Combinándolo con los comandos anteriores, podemos escar solo la información que deseamos de archivos log compuestos por cientos de líneas y de campos. Su uso es muy extenso, así que solo veremos varios ejemplos que nos ayuden a entenderlo. Algunas opciones básicas son:

- ▶ `-i`: Para que sea insensible a mayúsculas.
- ▶ `-n`: Nos muestra el número de línea.
- ▶ `-c`: Los resultados son los inversos a la expresión usada.
- ▶ `-v`: Interpreta el siguiente carácter como un si fuera normal y no como una expresión regular.

Viendo unos ejemplos nos resultará más fácil:

```
# grep '12345*'
# 12345*

# grep -i hola directorio/fichero*
# hola Mundo
```

Por supuesto, la potencia a la hora de filtrar la tenemos en las expresiones regulares.

## Variantes de Grep ▶

```
grep
fgrep
egrep
```

No vamos a explicarlas en este taller, pero más adelante se verán algunos ejemplos.

**■ AWK**

Ahora veremos un poco de `Awk`, que más que un comando es un lenguaje propio que nos permite filtrar, editar e imprimir con un gran nivel de exactitud. Requerirá su propio reportaje, así que simplemente vamos a ver unas cuantas nociones básicas y algunos ejemplos.

`Awk` interpreta toda la información en forma de campos, que por defecto se delimitan por un espacio, pero podemos definir el delimitador que nos interesa. Un ejemplo sencillo:

```
# echo hola Mundo Adios | awk
' {print $1 $3} '
# hola Adios
```

Dentro del propio `awk` podemos aplicar filtros, por lo que incluso es posible prescindir de `grep` en muchas ocasiones:

```
# cat /etc/passwd.conf |
awk '/^#aseeserver/{ print $2 }'
```

Como información podemos usar `print` para imprimir cualquier mensaje. Por ejemplo:

```
# cat /etc/passwd.conf | awk
'BEGIN{print "Servidor: "ENV["ip"]}
/^#aseeserver/{ print $2 } END{ }'
```

El resto es cuestión de jugar un poco.

**ANALIZANDO LOGS**

Ahora vamos a ver algunos ejemplos de cómo usar las anteriores herramientas para extraer información de un log. Vamos a seguir empleando archivos log genéricos de momento, y más adelante veremos otras herramientas para algunos concretos.

**LOGS MÁS ESPECÍFICOS**

Ya hemos visto sistemas para filtrar y entender logs genéricos o, mejor dicho, datos genéricos dentro de estos archivos. Ahora vamos a centrarnos en una serie de herramientas para logs concretos que los interpretarán por nosotros y nos darán el resultado de una forma que sea fácil de leer.

**■ WEBALIZER**

Esta aplicación permite realizar análisis estadísticos del uso de Internet basándose en los de `Squid` o `Apache`. `Webalizer` analiza estos logs y genera unos gráficos y paginas HTML que se pueden usar para interpretar fácilmente el uso de estas aplicaciones.

Realizar algo así sin una herramienta como ésta es poco más que una locura por la gran cantidad de información que genera este tipo de servicios.

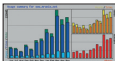
`Webalizer` está escrito en C y es extremadamente rápido. Puede procesar un archivo con unas 100.000 entradas (unos 45 Mb) en aproximadamente quince segundos. Aunque usualmente se utiliza para analizar estadísticas web, también puede ser usado para analizar logs de servidores FTP y del proxy-cache `Squid`. Una de sus características más interesantes es que soporta tanto archivos únicos de logs como los rotados.

Una vez instalado `Webalizer` en nuestro sistema, éste nos provee de un binario que debemos ejecutar cada cierto tiempo para que genere las imágenes y los HTML que podremos visualizar con nuestro navegador. Pero antes, nuestro servidor web, `Apache` en este caso, debe encargarse de loguear lo que necesitamos. Para ello debemos configurar `Apache` de la siguiente manera:

```
ErrorLog logs/WEBALIZER-error_log
TransferLog logs/WEBALIZER-access_log
LogFormat "%b %l %u %t \"%r\" %> %b
\"%{Referer}i\" \"%{User-agent}i\""
```

Y le diremos a `Webalizer` que analice esos dos archivos en `webalizer.conf`. Una configuración que nos puede ser útil es:

```
Incremental yes
VisitTimeout 1800
PageType htm
PageType html
PageType gif
PageType png
PageType jpg
IndexFile index.htm
IndexFile index.shtml
IndexFile index.gif
IndexFile index.png
SiteURL *.gif
SiteURL *.gif
SiteURL *.jpg
SiteURL *.png
SiteURL *.png
SiteURL *.png
SiteURL *.z
```



Ejemplo de gráfica generada por `Webalizer`.



resultado o mandarlo a una utilidad como Nagios.

Este comando resulta muy útil, ya que, por ejemplo, en lugar de usar los comandos que vimos al principio para analizar los logs y tener que recordar esa expresión regular, podemos crear dentro de asec y llamarla siempre que nos haga falta.

La instalación de asec es muy sencilla, pero su uso puede llegar a ser realmente complejo, debido a la gran flexibilidad que nos permite.

Necesitaríamos un reportaje entero para hablar de asec, pero simplemente sabiendo un poco sobre expresiones regulares ya conocemos el 95 por ciento de su funcionamiento.

Normalmente se crean archivos de reglas para analizarlos, así podemos llamar a asec con las reglas que nos interesen y filtrar un log. Algunos ejemplos se ven en el **Listado 1**.

Esto detectaría si la impresora ha sido abierta (por supuesto debe soportar este tipo de eventos) y nos enviaría un correo (ver **Listado 2**).

Analizaría el log señalado y nos mostraría en pantalla los logs que encuentre, indicando usuario y hora.

#### ■ LOGWATCH

Este comando es muy útil también y, como algunos de los anteriores, se debe ejecutar en el cron del sistema, por ejemplo cada diez minutos. Logwatch revisa los logs del sistema y nos envía un correo con un informe cuando encuentra algo sospechoso o desconocido.

Su uso es muy sencillo, basta con instalarlo y agregarlo al cron. Por defecto trae una configuración bastante adecuada, solo habría que especificar el correo del administrador en `/etc/logcheck/logcheck.sh`. El único problema con la configuración por defecto es que todo lo que no se haya especificado de forma explícita como un evento normal del sistema nos será enviado por correo, y esto puede resultar un poco pesado.

Para solucionarlo, basta con editar `/etc/logcheck/logcheck.ignore` y añadir expresiones de eventos que debe ignorar.

#### CONCLUSIÓN

Con todas estas herramientas y utilidades podemos tener un sistema vigilado y algo más seguro, ya que nos permitirá controlar aspectos que hasta ahora se nos pasaban por alto por no estar las 24 horas de día observando logs. Existen muchas más herramientas que nos pueden ser útiles, sobre todo a la hora de realizar análisis e informes automáticos de logs. ■

#### Listado 1. Ejemplos de reglas para el uso de Sec

```
type = single
desc = Informa de apertura de impresora $1
ptype = regex
pattern = ([^\. -]+) printer: cover/door open
context = ! expecting_cover_open_$1
action = delete Report_printer_$1 Offline; \
shellcmd /usr/bin/mailx -s \
"La impresora $1, ha sido abierta" admin@mil
```

#### Listado 2. Ejemplo de Código enviado por Sec al usuario

```
type=single
desc=Start login timer
ptype=regex
pattern=([A-Za-z0-9_-]+) sshd:[([0-9]+)\]: \[([*])+\] Accepted
(publickey|password) for ([A-Za-z0-9_-]+) from [0-9.]+ port [0-9]+ (.*)
action= shellcmd echo "session_log_$1_$2" "session_log_owner_$1_$2"
```

#### Referencias ▶

- ▶ Syslog (syslogd): <http://www.infodrom.org/projects/syslogd/>
- ▶ Syslog-ng: [http://www.balabit.com/products/syslog\\_ng/](http://www.balabit.com/products/syslog_ng/)
- ▶ Metalog: <http://metalog.sourceforge.net/>
- ▶ Logrotate: <http://laln.cx/irc/logrotate/>
- ▶ Webalizer: <http://www.mnux.net/webalizer/>
- ▶ Calamaris: <http://cord.de/tools/squid/calamaris/Welcome.html.en>
- ▶ Proxy-cache squid: <http://www.squid-cache.org/>
- ▶ Denyhost: <http://denyhosts.sourceforge.net/>
- ▶ Sec: <http://simple-evtcorr.sourceforge.net/>
- ▶ Logwatch: <http://www.logwatch.org/>



EL SOFTWARE LIBRE  
HA LLEGADO AL OTRO LADO

WWW.ELOTROLADO.NET

NOSOTROS TAMPOCO GUSTAMOS A MICROSOFT