

## Head, Tail, Cat

# De Cabo a Rabo

Pequeñas utilidades como `head`, `tail` y `cat` se usan para mostrar el contenido de los ficheros de texto. Otros comandos y programas proporcionan funciones parecidas. Aunque todos traten con ficheros de texto, estas herramientas de línea de comando son dignas de mención ya que pueden combinarse para formar potentes herramientas.

POR HEIKE JURZIK



Linux proporciona unos cuantos programas capaces de mostrar ficheros de texto. Por supuesto, se puede arrancar un editor como Vi ó Emacs ó un paginador como `less` o `more`, para leer estos ficheros. Pero ¿por qué no usar algo más simple, como `cat`, `head` y `tail`?

## El Gato Multifunción

El programa `cat` no sólo “concatena” múltiples ficheros para crear uno sólo, también es útil para mostrar ficheros individuales en la **salida estándar** del equipo. De la forma más simple, `cat fichero.txt`, muestra por pantalla el contenido de `fichero.txt`.

A veces se necesita añadir números de línea a un fichero de texto para referirse a estas líneas. Esto, por ejemplo, puede facilitar la explicación del listado de un

programa. Para hacerlo, `cat` usa la opción `-n` cuya salida coloca un número al principio de cada línea del fichero.

Pero hay que tener cuidado con los formatos binarios. Si por error se muestra un fichero binario usando `cat`, no sólo podríamos encontrarnos con una pantalla llena de caracteres de control extraños, sino que las propias entradas podrían volverse ilegibles (ver Figura 1). El comando `reset` puede ayudar a remediar esta situación. Escribimos `reset` en el prompt, y presionamos [Return] para restaurar el terminal. Puede que no seremos capaces de ver lo que escribimos. Si esto no ocurre también puede intentarlo con `echo [Ctrl-v][Esc][c][Return]`.

`cat` se usa a menudo para redireccionar datos mediante una **cauce** a otros programas. Se puede usar `cat` como salida a

una lista que será redireccionada al comando `sort`:

```
$ cat lista.txt | sort
Arnie 42
Easter 120
Petronella 100
Spátula 30
```

Cualquier salida desde `cat` puede redireccionarse fácilmente usando herramientas típicas de Unix. El operador `>` envía la salida a un nuevo fichero, `fichero2`:

```
cat fichero1 > fichero2
```

Esto se puede hacer con múltiples ficheros. Para hacerlo así, simplemente hay que teclear el nombre de los ficheros uno tras otro:

```
cat fichero1 fichero2 > fichero3
```

Si `fichero3` no existe, este comando lo creará. Si el fichero existe, `cat` simplemente lo sobrescribirá. Un operador distinto permitirá a `cat` añadir a un fichero existente:

## Línea de Comandos

Aunque los GUIs (interfaces de usuario) como KDE o GNOME son útiles para muchas tareas, si se tiene la intención de dominar al máximo una máquina Linux, necesitaremos volver a utilizar la antigua línea de comandos de vez en cuando. Además de esto, probablemente nos enfrentaremos a varios escenarios donde algún conocimiento básico será muy útil al poder movernos por la jungla de los comandos de la shell.

### Listado 1: head incluye un separador para múltiples ficheros

```
01 huhn@asteroid:/etc$ <B> head
   -n 3 *.conf
02 ==> aatv.conf <==
03 # fichero de configuración de
   aatv.
04 # Modo de video. Escoger entre
   PAL, SECAM y NTSC.
05 ==> abcde.conf <==
06 # Sistema por defecto para
   abcde.
07 # Nada en este fichero está
   descomentado por defecto.
08 #
09 ==> adduser.conf <==
10 ...
```

### Listado 2: Combinación de tail -f y grep

```
01 root@marvin[~] <B>tail -f
   /var/log/isdn/isdnlog | grep
   RING<B>
02 Sep 25 13:25:51 * Call to tei
   127 from +353 77777777,
   Ireland on +44 207/40, London
   RING (Speech)
03 Sep 25 13:31:09 * Call to tei
   127 from +39 2/44444444, Milano
   on +44 207/40, London RING
   (Speech)
04 Sep 27 12:38:30 * Call to tei
   127 from ? on + 44 207/40,
   London RING (3.1
05 kHz audio)
```

```
cat fichero1 fichero2 >>>
fichero3
```

Si redirigimos la salida a un fichero de esta forma, *cat* no tendrá problemas en el manejo de ficheros binarios. Esto permite el uso de *split* para distribuir un

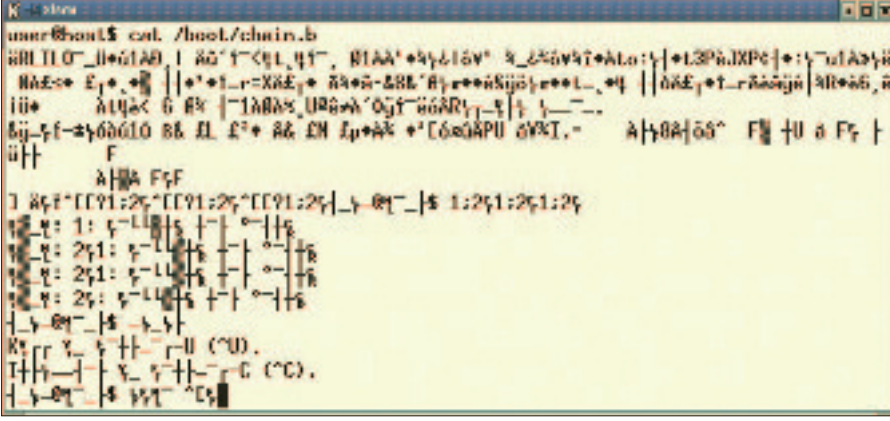


Figura 1: “Basura binaria” en la línea de comandos – resetear normalmente ayuda.

fichero muy largo a través de múltiples paquetes que son reensamblados usando *cat*. Una aplicación típica de esta técnica sería para transportar datos en disquetes o en ficheros adjuntos en los emails.

Algunos grupos de noticias, como *comp.unix.shell* han estado realizando un concurso, desde hace unos años, llamado “Useless Use of Cat Award” [1]. El ganador es la persona que sugiera la forma más inútil de usar el comando *cat*. La idea es la siguiente: “El propósito de *cat* es concatenar ficheros. Si un fichero, concatenándolo con nada, es una pérdida de tiempo y coste, tenemos lo que buscamos”. Dejo a la discreción de los lectores lo que consideren “inútil”. Pero un uso redundante de *cat* sigue el principio de Unix del uso de pipes para conectar las herramientas pequeñas. Y cualquier hardware moderno debería ser capaz de manejar ese proceso adicional sin demasiado esfuerzo.

### A la Cabeza

*Nomen est omen*: *head* muestra la primera línea de un fichero por pantalla. Si se ejecuta *head fichero.txt* normalmente se mostrarán las diez primeras líneas. Se puede cambiar esta opción que trae por defecto usando *-n número*. El fichero de

configuración en */etc* a menudo indica el nombre y uso del fichero en la primera línea. Para mostrar las tres primeras líneas de los ficheros de configuración de este directorio se puede teclear *head -n 3 \*.conf*

Práctico: Si se ejecuta *head* para múltiples ficheros, la herramienta añade un separador visible (Listado 1). Si no quiere que aparezca este separador se puede utilizar el parámetro *-q* (*--quiet*).

### Final Feliz

El comando *tail* trabaja de forma similar, pero como sugiere el nombre, la salida es la última línea de un fichero. Se pueden especificar múltiples ficheros en la salida. Junto con la opción *-n número*, la opción *-f* es muy usada. Esto permite visualizar las últimas líneas de ficheros a los que están continuamente añadiéndose datos, como los ficheros de logs. Así *tail* para un fichero de logs de RDSI indicará quién está conectado a través del teléfono (Listado 2).

Se puede obtener el mismo efecto con el paginador *less*, simplemente tecleando [Shift+f] mientras se visualiza el fichero. El programa entonces escuchará las nuevas líneas en el fichero (aparece el estado de las líneas leídas:

```
Waiting for data
...(interrupt to abort)
```

mientras se hace). Tecleamos [Ctrl-c] para salir del modo “Follow”.

### GLOSARIO

**Salida estándar:** Hay tres “canales estándar” para la entrada y la salida, *stdin* (entrada estándar), *stdout* (salida estándar) y *stderr* (salida estándar de errores). La entrada estándar típicamente es el teclado, mientras que la salida y el error estándar normalmente usarán la pantalla. Los operadores especiales *>* y *<* permitirán redireccionar

estos canales de E/S para enviar la salida de error estándar a un fichero, por ejemplo. **Pipe:** Permite encadenar múltiples comandos conectando la salida de un comando con la entrada de otro. Los pipes se crean usando el carácter *|*. Como *ls-l | less* enviará la salida del comando *ls* al paginador *less*, el cual lo mostrará por pantalla página a página.

### RECURSOS

[1] Useless Use of Cat Award: <http://rhols66.adsl.netsonic.fi/era/unix/award.html#cat>