Artículo 6: Como utilizar MUTT

Autor: (c) Santiago Romero
Revista: Solo Linux (Prensa Técnica) - Diciembre-2000

Mutt es posiblemente el cliente de correo más famoso y más utilizado bajo Linux, junto con PINE. Consume una mínima cantidad de recursos, funciona en consola y admite colores y personalización por parte del usuario, se integra con PGP y GnuPG y se pueden especificar las combinaciones de teclas deseadas para su uso.

Mutt es uno de los mejores clientes de correo existentes para Linux, al disponer de un perfecto equilibrio entre funcionalidad y consumo de recursos, y por ser muy sencillo de utilizar. Hay diferentes versiones (está la versión internacional, que lleva una 'i' en el paquete rpm/deb y la US, que no permite el uso del PGP ya que en USA está prohibido). Disponemos también de diferentes versiones estables (1.0 y 1.2) y otras de desarrollo, aconsejándose el uso de una versión estable si lo vamos a utilizar para uso diario.

En cualquier caso deberemos instalar el paquete adecuado (mutt-version o mutt-version-i). Para ello bastará con montar el CD de nuestra distribución de Linux e instalar el paquete con rpm o dpkg (según usemos distribuciones basadas en uno u otro sistema de paquetes), o descargarlo de sitios como http://rpmfind.net o http://www.debian.org e instalarlos. Los usuarios de Debian que dispongan de una conexión a Internet pueden utilizar apt-get install slrn para instalar la última versión estable descargada automáticamente del site de Debian.

Tras la instalación del programa deberemos proceder a su configuración, aunque es cierto que los valores del programa por defecto permiten comenzar a utilizarlo inmediatamente a su instalación debido a su sencilla ayuda en línea en la barra de estado de la pantalla. Antes de entrar en el mismo (el nombre del ejecutable es "mutt") deberemos crear un directorio Mail (donde meteremos nuestro correo de entrada y de salida, así como los mensajes de las listas de correo):

```
mkdir /home/usuario/Mail
```

Mutt nos permitirá leer el correo disponible en el spool de Linux. Para aprovechar esto y para poder utilizarlo offline (es decir, leer y contesar el correo con el modem desconectado) deberemos tener correctamente configurados sendmail y fetchmail.

FICHERO DE CONFIGURACION .MUTTRC

Al igual que vim, la configuración de mutt se realiza mediante un fichero de configuración en el directorio HOME del usuario que lo ejecutará, llamado .muttrc. Editando este fichero e insertando ahi las opciones que deseemos para el programa es como cada usuario personaliza mutt para su uso personal, adecuándolo a sus necesidades.

El hecho de configurar MUTT utilizando un fichero de opciones permite tener accesibilidad a todos y cada uno de los parámetros configurables de MUTT y cambiar gran parte de los colores, teclas y menues del programa mediante la edición de un simple fichero de texto. Podemos obtener versiones de ejemplo de dicho fichero en /usr/doc/mutt/examples/sample.muttrc, de forma que baste con copiar dicho fichero como .muttrc y adaptar las opciones que lleve por defecto a nuestras propias necesidades. Dichas opciones están convenientemente comentadas en el fichero de ejemplo y en la documentación disponible en /usr/doc/mutt.

En el listado 1 puede verse un ejemplo de fichero .muttrc.

```
Fichero de configuración:  .muttrc
-----------------------------------------------------------------

set folder=~/Mail               # Dónde guardar todo.
#set spoolfile='~/Mail/inbox'   # De dónde leer el correo nuevo.
                                   Si usais procmail descomentad esto
                                   y dirigidlo a vuestro mbox.
set mbox=+read                  # Dónde guardar los mensajes leídos.
set record=+sent_mail           # Donde guardar mensajes de salida.
set move=yes                    # Mover mensajes sin preguntar.
set copy=yes                    # Copia mensajes salientes.
source ~/.mail_aliases          # Incluir mail aliases.
set abort_nosubject=ask-yes     # Preguntar "Abortar msg" si vacío.
unset autoedit                  # No ir directo al editor.
set auto_tag                    # Aplicar comandos a msgs marcados.
set nobeep                      # No pitar en errores.
set charset="iso-8859-1"
set locale="es_ES"
set allow_8bit                  # (las 3 anteriores) Castellano.
set editor="/usr/bin/vim +8"    # Usar VIM empezando en línea 8.
set edit_headers                # Permitir cambiar cabeceras.
set hdrs
set fast_reply                  # No preguntar cabeceras (usar From)
unset force_name                # No grabar en carpetas con nombre.
set forward_format="Re: %s"     # Formato de las respuestas.
set forward_quote               # Incluir texto quoteado en forwards
set realname="Santiago Romero"  # Nombre para el From.
set hostname="iname.com"        # Para el From: @
set include                     # Incluir texto quoteado.
set indent_string="> "          # Cadena de quoteado.
unset metoo                     # No autocontestarme.
unset mime_forward_decode
set postpone=ask-yes            # Preguntar si quiero postponer msgs.
set print=ask-no                # Preguntar si deseo imprimir el msg.
```

```
set delete=yes                  # Borrar mensajes sin preguntar.
set print_command=/bin/false    # No imprimir.
set noprompt_after
set quit=ask-yes                # Preguntar si quiero salir.
set recall=no
unset save_name
set shell="/bin/sh"             # Shell que queremos usar.
set sig_dashes                  # Anteponer -- a firma.
set signature="~/.signature"    # Fichero de firma
set sort=threads                # Método de ordenación principal.
set sort_aux=last-date          # Método de ordenación auxiliar.
#set sort_browser=reverse-date  # Método de ordenación de ficheros.
set tmpdir=/home/yo/Mail/tmp    # Directorio temporal (tb puede ser /tmp).
set nowait_key
#folder-hook . "push <esc>V"    # Colapso de los threads automático.
mailboxes `echo $HOME/Mail/*`
hdr_order date from reply-to subject to cc
auto_view application/x-gunzip
lists valux-list lista-eya      # Listas de correo que tenemos.
#set pager_index_lines=10       # Configuración del pager.
set pager_stop                  # No avanzar al ste mensaje...
bind pager <up> previous-line
bind pager <down> next-line
bind pager p previous-undeleted
bind pager n next-undeleted
color hdrdefault red default    # Colores de MUTT similares a slrn
color quoted cyan default
color signature red default
color indicator brightyellow red
color error brightred default
color status yellow blue
color tree white default
color tilde magenta default
color message brightcyan default
color markers brightcyan default
color attachment white default
color search yellow blue
color header brightred default ^(From|Subject):
color body magenta default "(ftp|http)://[^ ]+"
color body magenta default [-a-z_0-9.]+@[-a-z_0-9.]+
color underline brightgreen default
unset use_from
unmy_hdr From:
my_hdr From: Santiago Romero <compiler@iname.com>
set quote_regexp="^ *[a-zA-Z]*[>:#}]"
set reply_regexp="^(re|aw):[ \t]*"
set date_format="%a, %d de %b de %Y, a las %I:%M:%S%p %Z"
set attribution="El %d, %n dijo:"
```

A continuación vamos a comentar algunas de las lineas:

```
set spoolfile='~/Mail/inbox'   # De dónde leer el correo nuevo.
```

En principio el correo nuevo se lee de /var/spool/mail, que es donde lo dejan programas como fetchmail o animail. Si nosotros no usamos procmail para la lectura del correo es aconsejable no incluir esta línea y dejarle a mutt el valor por defecto de spoolfile. Si en cambio usamos procmail de forma que nos deja

todo el correo nuevo en (por ejemplo) un fichero Mail/inbox, podemos especificarle a mutt este fichero, conocido como Bandeja de Entrada en otros sistemas. Para aclarar esto basta decir que cuando un programa de Linux recoge el correo lo deja en un sólo fichero que contiene (uno tras otro) todos los emails recibidos. Si no usamos ningún tipo de clasificador de correo todos los mensajes recibidos por programas como fetchmail irán a parar a /var/spool /mail/nuestro_usuario. Si por contra tenemos algún clasificador de correo (como procmail, que permite distribuir el correo recibido a diferentes ficheros según el tema, remitente, etc.) podemos especificarle a mutt en qué fichero deja procmail el nuevo correo, de forma que cuando arranquemos mutt aparezca directamente este fichero como buzón de entrada, y veamos los mensajes contenidos en él.

Es importante que en nuestro fichero de reglas de procmail aparezca una regla que indique a nuestro MTA que debe dejar el correo en el mismo lugar en que le decimos a MUTT que debe buscar, para que éste encuentre fácilmente todos los mensajes. Siempre es posible no usar procmail y decirle a mutt que lea el correo de /var/spool/mail/usuario, aunque con esto perdermos las funcionalidades de procmail.

```
set editor="/usr/bin/vim +8"    # Usar VIM empezando en línea 8.
```

Esta línea nos permite usar VIM como el editor de texto estándar para mutt, indicandole además que empiece a editar el fichero (el e-mail) en la línea 8 del mismo, justo tras las cabeceras y antes de la firma del mensaje. De este modo podemos utilizar cualquier editor con MUTT. El editor es llamado cuando respondemos un mensaje, y cambiando esta opción podremos utilizar aquel editor al que estemos más acostumbrados, como Kwrite, Pico, mcedit, gedit, Vim, joe o similares.

```
lists valux-list lista-eya      # Listas de correo que tenemos.
```

Mediante el comando lists le podemos especificar a mutt los nombres de ficheros que contienen los mensajes recibidos desde listas de correo. Recordemos que normalmente se usan clasificadores de correo como procmail para separar el correo en diferentes carpetas. Un uso habitual de procmail es para separar el correo normal (que se suele dejar en un fichero inbox o de nombre similar) de los correos recibidos desde listas de correo, que se dejan en ficheros aparte para no mezclar el correo diario con las listas. Con lists podemos especificar los nombres de estos ficheros para que posteriormente podamos enviar mensajes a las mismas usando la tecla 'L'. Esta opción está disponible únicamente para las últimas versiones de Mutt. Si nuestra versión es antigua y al arrancar Mutt obtenemos un error de comando no reconocido, deberemos eliminar esta línea.

```
color hdrdefault red default    # Colores de MUTT similares a slrn
color quoted cyan default
color signature red default
```

```
color indicator brightyellow red
color error brightred default
color status yellow blue
color tree white default
color tilde magenta default
(etc..)
```

Este tipo de opciones nos permitirán configurar los colores de mutt a nuestro antojo. En este caso se le ha dado un look similar a SLRN, un lector de noticias de Linux, el cual veremos en la próxima entrega.

```
my_hdr From: Santiago Romero <compiler@iname.com>
```

Mediante esta opción podemos definir la línea From: que queremos que aparezca en los mensajes, diseñándola a medida de nuestras necesidades (para poder colocar cadenas anti-spam, nuestro nombre, etc.). En este caso pondremos nuestra dirección real, la que queremos que salga como From de nuestros mensajes. Gracias a esta opción de Mutt podemos cambiar las cabeceras con que salen nuestros mensajes del sistema y no necesitaremos que el servidor de correo (sendmail, postfix o exim) enmascare los mensajes salientes para cambiar el dominio de nuestra máquina por el dominio correcto de cada usuario.

RESUMEN DE TECLAS DE MUTT

En la tabla 1 podemos ver un resumen de teclas de mutt.

```
  Tecla       Función
-------------------------------------------------------------------------
  m           Componer nuevo mensaje.
  r           Responder mensaje actual.
  L           Responder a la lista de correo.
  d           Borrar mensaje actual.
  u           Recuperar mensaje borrado.
  C           Copiar mensaje a carpeta.
  s           Salvar mensaje/attach a disco y marcarlo como borrado.
  c           Abrir buzon/carpeta.
              =nombre  ->  Abrir /home/sromero/Mail/nombre
              ?        ->  Ver lista de buzones.
              !        ->  Mail spool.
  N           Marcar mensaje como no leído.
  a           Añadir alias de correo (en .mail_aliases).
              (o bien adjuntar ficheros en la ventana de envío).
  ENTER       Ver mensaje.
  v           Ver attachments del mensaje (se podrán grabar con s).
  q           Salir de MUTT o del visualizador de mensajes.
  y           Confirmar envío del mensaje.
  t           Marcar mensajes.
  ;           Decirle a mutt que la próxima acción se realice sobre
              todos los mensajes marcados (para borrar, salvar, etc).
  $           Eliminar inmediatamente los mensajes marcados.
  /           Buscar mensajes.
```

```
/ ~b        Buscar dentro del cuerpo.
```

De estas teclas cabe simplemente destacar (ya que mutt es un programa de muy sencillo uso y con ayuda online) las siguientes teclas:

```
m           Componer nuevo mensaje.
r           Responder al mensaje actual.
d           Borrar mensaje actual.
u           Recuperar mensaje borrado.
s           Salvar mensaje/attach a disco y marcarlo como borrado.
c           Abrir buzon/carpeta.
N           Marcar mensaje como no leído.
ENTER       Ver mensaje.
q           Salir de MUTT o del visualizador de mensajes.
```

Mediante el uso de apenas 10 teclas podremos realizar cualquier tarea básica de lectura, edición y envío de correo con un gasto mínimo de recursos. No es necesario abrir enormes programas (como Netscape Communicator) para realizar tareas tan sencillas como las que realiza correctamente mutt.

La mayoría de teclas de Mutt son reconfigurables, por lo que no es necesario aprender las teclas de uso de Mutt si estamos acostumbrados a otros clientes de correo. Basta con editar el fichero de configuración de mutt (.muttrc) y añadir líneas similares a las siguientes:

```
bind pager  previous-line
bind pager  next-line
bind pager p previous-undeleted
bind pager n next-undeleted
```

El comando bind sirve para asignar una tecla a un evento dentro de una parte determinada del programa. La sintaxis es:

```
bind lugar tecla acción
```

En la primera de las línas le decimos a Mutt que queremos añadir la tecla CURSOR ARRIBA como función de Ir a la Anterior Línea en el pager (el visualizador de mensajes que viene incluido con mutt). En realidad estás 4 redefiniciones de teclas se han realizado para asignar a mutt teclas similares a las que usa SLRN,donde se usan los cursores para moverse por los mensajes, y P y N para moverse al siguiente y anterior mensaje no leído.

UTILIZANDO MUTT

Al Entrar en mutt (con nuestro .muttrc correctamente creado) veremos que hay varios mensajes, como en el ejemplo de la figura 1 (si los hemos recogido con fetchmail).

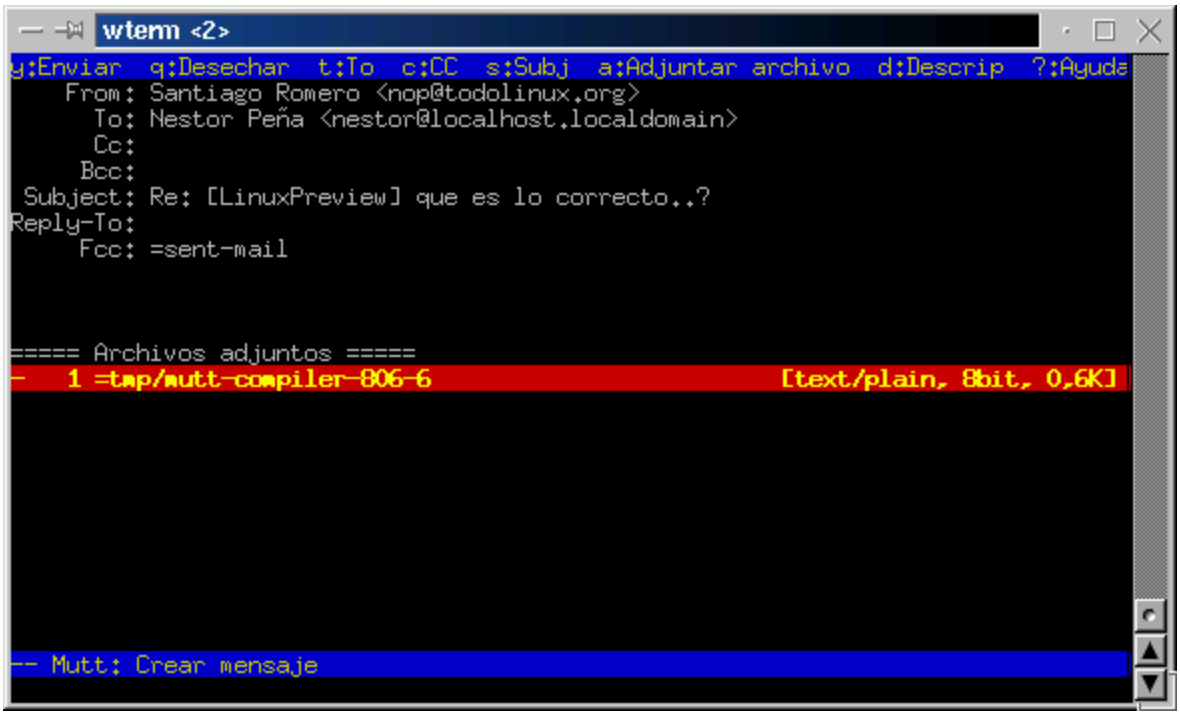
Si no tenemos mensajes y queremos probar mutt sin miedo a perder mensajes mientras aprendemos a usarlo, podemos auto-enviarnos mensajes como usuarios o desde root con el comando mail:

```
mail -s "titulo" usuario
```

Con este comando los mensajes se finalizan con un '.' al principio de la línea final del mensaje). Utilizando este comando podemos enviar múltiples mensajes desde un usuario a otro o desde root a un usuario para practicar en el uso de mutt sin temor a perder los mensajes del correo habitual (mientras recogemos estos con el cliente de correo que utilicemos normalmente). Cuando ya se posea un poco de práctica con mutt se podrá recoger el correo real directamente con él y trabajar ya con el correo bajo este estupendo programa.

Cuando hay varios mensajes podemos desplazarnos hasta el deseado con las teclas del cursor y pulsar Intro para leerlo. Una vez dentro del pager nos movemos con las teclas de arriba y abajo para su completa lectura. Si no deseamos contestarlo, salimos del pager (que además podemos sustituir por cualquier programa) con la tecla 'q' y apareceremos de nuevo en la lista de mensajes. Con 'r' (reply) nos aparecería el editor por defecto y podríamos responder el mensaje (saliendo finalmente de vim con ESCAPE : x). El mensaje normalmente incluirá el texto del mensaje anterior quoteado con el carácter > para permitirnos su edición y modificación.

Tras grabar la respuesta (con el nombre de fichero que haya por defecto, sin modificarlo) aparecerá una nueva ventana (figura 4) donde podemos enviar el mensaje ('y'), desecharlo ('q'), añadir ficheros/attachments ('a') o reeditarlo ('e').



Si pulsamos 'a' (Añadir Adjunto) podremos incorporar al mensaje otros documentos como ficheros de texto, ficheros binarios, imágenes, etc, de forma que aparezcan al destinatario como adjuntos al mismo. Estos adjuntos (en el caso de que seamos nosotros los destinatarios, y nos llegara un mensaje con ficheros binarios incorporados) se pueden grabar pulsando 'v' y 's', para abrirlos con el programa deseado, o dejar que lo abra mutt directamente con el programa que hayamos asociado a dicha extensión.

En el caso de pulsar 'y' para enviar el mensaje apareceremos de nuevo en la ventana de mensajes. Habrá desaparecido la N que lo marcaba como nuevo o la O que lo marca como OLD (viejo) y en su lugar aparece una r (de respondido). Si nos interesa borrarlo (ya que lo hemos respondido, y dado que los que no se borran se guardan en Mail/read) sólo tenemos que pulsar la tecla 'd' (de delete). El mensaje se quedará marcado para borrado y será eliminado cuando abandonemos el programa, opción que se puede abortar mediante Undelete (tecla 'u').

Cuando leemos un mensaje que queremos seguir conservando sin que vaya a la carpeta de read lo marcamos pulsando 'N' (nuevo) y para mutt será como si no lo hubiéramos leído (apareciendo en esa misma "carpeta" como un mensaje nuevo en la próxima consulta).

Si queremos conservar un mensaje o bien moverlo a un fichero/carpeta lo

podemos hacer con 'C' (copiar) o 's' (mover = copiar y borrar). Con 'c' (minúscula) podremos abrir los ficheros del directorio Mail como si fueran buzones de mensajes , y navegar por ellos, visualizándolos, etc. Esto permite navegar por el correo respondido (sent-mail), por el leído (read), por los mensajes que hayamos guardado previamente (lista-correo-tal), etc. Además si estamos usando PROCMAIL para que el correo de las listas de correo vaya a determinado fichero o carpeta, podremos trabajar también con este tipo de listas, y mantener nuestro correo personal separado de las listas de correo. Esto es realmente cómodo y permite interconectar las habilidades de procmail con las de mutt y las de SLRN, como veremos en la próxima entrega.

MAIL ALIASES

Los mail aliases vienen a ser el equivalente en mutt a las libretas de direcciones en otros programas de correo. Supongamos que tenemos un amigo (Jose) al que escribimos muy asíduamente. Su email es emaildejose_eslargo@eslargo.es. ¿Cada vez que le queremos escribir un email debemos teclearlo todo? No: podemos crear alias de correo. Para ello en mutt nos ponemos sobre un email de dicha persona (en la lista de mensajes) y le damos a la tecla 'a' (de alias). Contestaremos las preguntas que nos hace y le diremos que grabe los cambios en el fichero .mail_aliases (la última de las preguntas). A partir de ese momento cuando queramos crear un correo nuevo para Jose, en el TO: sólo hemos de poner Jose (o lo que hayamos contestado al comando 'a') para que mutt cambie su nombre por su dirección correcta. De este modo tenemos nuestra libreta de direcciones de acceso rápido. Este fichero de direcciones puede además ser editado directamente por el usuario, y es estándar para otros programas, con un formato similar al siguiente:

```
alias Jose  Mi amigo <jose@jose.es>
alias Jefe  Jose Luis <direccion@correo.es>
alias Sonia Sonia Fernandez <sonia@nov.es>
(etc.)
```

En general:

```
alias ALIAS NOMBRE <DIRECCION>
```

Mutt tiene muchas más funcionalidades (autofirmado pgp, etc.) pero sólo se han comentado las más comunes para su funcionamiento básico, ya que conforme el usuario vaya descubriendo las diferentes funcionalidades, descubrirá la verdadera potencia de Mutt.

Por ejemplo: para integrar GnuPG en mutt deberíamos incluir las siguientes líneas en el .muttrc:

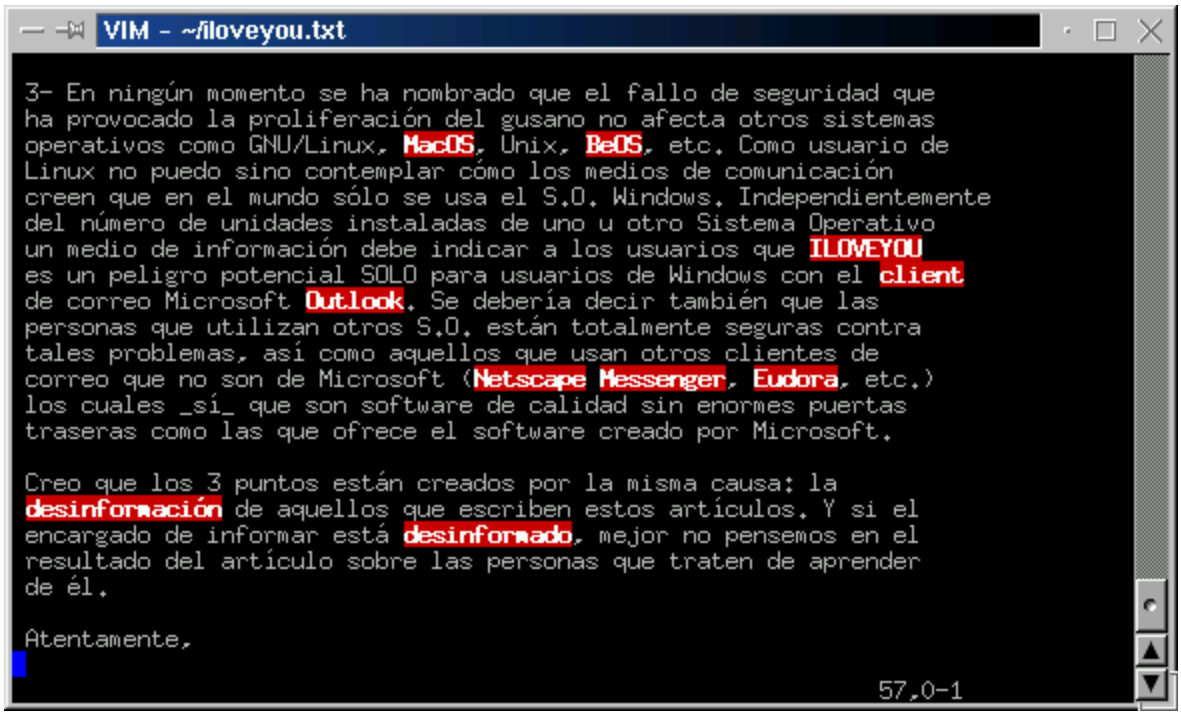
```
set pgp_replysign
set pgp_verify_sig=yes
set pgp_default_version=gpg
```

```
set pgp_gpg=/usr/local/bin/gpg
set pgp_key_version=default
set pgp_receive_version=default
set pgp_send_version=default
set pgp_sign_micalg=pgp-sha1
```

Para poder integrar PGP o GPG en mutt es necesario que dispongamos de la versión Internacional (i) del programa, pues en los E.U.A la tecnología de cifrado está limitada y las versiones US del programa tienen algunas pegas a la hora de trabajar con firmas y mensajes cifrados. La utilización de PGP y GPG se está extendiendo enormemente ante la concienciación de las personas de la importancia de la privacidad de los mensajes, sobre todo de algunos con determinados contenidos personales. Si el lector desea profundizar en el uso de herramientas de cifrado y leer más información sobre la materia puede acudir a los diferentes documento disponibles en http://www.kriptopolis.com .

CORRECION ORTOGRAFICA CON VIM

Aprovechando la infinitas posibilidades de Vim podemos utilizar correción ortográfica en nuestros documentos de texto y en nuestros emails contestados con mutt (y news contestadas con SLRN). Para ello lo que haremos será dotar a Vim de interactividad con ispell, el paquete de corrección ortográfica de Linux, instalando además un diccionario de palabras en castellano, lo cual nos dará la oportunidad de corregir textos en castellano y en inglés. La corrección será no interactiva, es decir, no tendremos que ir palabra por palabra reemplazando los errores sino que se marcarán en rojo las palabras que no estén en el diccionario de ispell (un diccionario ampliable por el usuario), como puede apreciarse en la figura 3.



Para el uso de esta funcionalidad necesitaremos tener instalado el corrector ortográfico ispell (disponible en la mayoría de distribuciones en formato nativo) junto con el diccionario castellano de ispell (espa~nol.tar.gz o bien ispell-spanish). El fichero-diccionario espa~nol.tar.gz lo podemos obtener en la página Web http://www.datsi.fi.upm.es/~coes/espell_leame/espell_leame.html , que además contiene otros enlaces relacionados con la corrección ortográfica, como un enlace directo al programa ispell. En ciertas distribuciones es posible que ya tengamos instalados los diccionarios, porque a veces se incluyen como un paquete llamado ispell-spanish (basta cerciorarse mediante "rpm -qa | grep ispell"). También hay otros diccionarios disponibles (catalán, etc.).

Para ello primero de todo creamos (con un editor de texto como vim) un fichero /usr/local/bin/vimspell.sh (o en cualquier otro lugar, siempre que luego seamos consecuentes dentro del script) tal y como puede verse en el listado 2.

```
#!/bin/sh
#
# Spell a file & generate the syntax statements necessary to
# highlight in vim.  Based on a program from Krishna Gadepalli
# .
#
# I use the following mappings (in .vimrc):
#
#       noremap  :so `vimspell.sh %`
#       noremap  :syntax clear SpellErrors
#
# Neil Schemenauer
# March 1999
```

```
INFILE=$1
OUTFILE=/tmp/vimspell.$$
# if you have "tempfile", use the following line
#OUTFILE=`tempfile`

#
# local spellings
#
#LOCAL_DICT=${LOCAL_DICT-$HOME/local/lib/local_dict}

if [ -f $LOCAL_DICT ]
then
        SPELL_ARGS="+$LOCAL_DICT"
fi

spell -despa~nol -Tlatin1 < $INFILE | sort -u |
awk '
      {
        printf "syntax match SpellErrors \"\\<%s\\>\"\n", $0 ;
      }

END   {
        printf "highlight link SpellErrors ErrorMsg\n\n" ;
      }
' > $OUTFILE
echo "!rm $OUTFILE" >> $OUTFILE
echo $OUTFILE
```

Tras teclear el listado le debemos dotar de permisos de ejecución mediante:

```
 chmod +x / usr/local/bin/vimspell.sh
```

Tras esto editamos nuestro fichero .vimrc y agregamos las siguientes macros de teclado para VIM (que nos permitirán llamar al anterior script y realizar la corrección):

Para textos normales:<

```
noremap  :so `/usr/local/bin/vimspell.sh %`
noremap  :syntax clear SpellErrors
```

Para usar con VIM en edición de emails/news (hay que grabar antes de corregir):

```
noremap  :w:so `/usr/local/bin/vimspell.sh %`
noremap  :syntax clear SpellErrors
```

Podemos usar uno u otro conjunto de macros en función del tipo de uso que le demos a Vim, o bien podemos también usar ambos conjuntos cambiando las teclas para alguno de ellos. El hecho de que haya 2 tipos de macros se deriva de que para corregir un fichero con nuestro anterior script es necesario que el fichero YA esté grabado en el disco, mientras que durante la edición de un email

no se graba hasta que salimos con :x del mismo, momento en que se graba y lo toma mutt para su envío. Mediante el segundo conjunto de macros se realiza un :w (grabar) antes de llamar al corrector, para que pueda corregir el fichero.

Finalmente nos hacemos con el paquete espa~nol.tar.gz (un diccionario de castellano con gran cantidad de vocablos) y lo descomprimos, ejecutamos make, y copiamos cada fichero a su lugar correspondiente (el make tarda bastante y necesita al menos 100Mb de espacio temporal que luego libera, con el fin de crear un ficherito que será el que usaremos). También podemos encontrar versiones ya compiladas que ocuparán mucho menos espacio que el requerido durante la compilación:

```
cp espa~nol.aff /usr/lib/ispell
cp espa~nol.hash /usr/lib/ispell
cp espa~nol.words /usr/dict
```

Este último paso no sería necesario en caso de disponer ya de los ficheros de diccionario preinstalados.

Por otra parte, un método de uso interactivo de ispell en la línea de comandos (ya fuera del uso dentro del editor VIM) sería el siguiente:

```
ispell -despa~nol -Tlatin1 fichero.txt
```

Si queremos simplemente obtener una lista de errores que aparecen en el texto, podemos ejecutar lo siguiente:

```
ispell -despa~nol -Tlatin1 < fichero.txt
```

Finalmente cabe decir que podemos añadir palabras a nuestro diccionario en nuestro home (.ispell_espa~nol o .ispell_words). Es una simple lista de palabras (ordenadas alfabéticamente con sort) con la primera línea en blanco. También podemos añadirlas mediante la ejecución interactiva de ispell, que dispone de una opción para añadir palabras no reconocidas al diccionario personal del usuario que lo ejecuta. Además podemos crear otro vimspelleng.sh con otras opciones y otra macro para VIM para corregir textos en inglés, si es que los escribimos habitualmente.

FINALIZANDO

La potencia de Linux radica en la interconexión de herramientas. En nuestra anterior entrega aprendimos a usar Vim y vimos que era un excelente editor todo-terreno. Hoy hemos visto cómo es posible usar Vim dentro de mutt (y como veremos, dentro de cualquier otro programa) de modo que todas las funcionalidades que tenía Vim pasan ahora a estar incluidas en mutt. Si tenemos una macro muy útil para Vim, la podremos usar para contestar correo,

por ejemplo.

En nuestra próxima entrega instalaremos y configuraremos SLRN, uno de los mejores clientes de noticias existente para Linux. Al igual que mutt, presenta todas las funcionalidades que cabe esperar de un cliente de news pero de una forma compacta y funcional desde consola. Y es que con programas como Vim, Mutt y Slrn, tener Linux en un 386 ó 486 puede dar mucho de sí para realizar todas las tareas habituales relacionadas con Internet.

Santiago Romero

Volver a la tabla de contenidos.