Mutt-i, GnuPG e PGP Howto

Andrés Seco *AndresSH@ctv.es* e J.Horacio M.G. *homega@ciberia.es* v.12, Febbraio 2000

Questo documento spiega brevemente come configurare *Mutt-i*, *PGP* nelle sue diverse versioni (2.6.x, 5.x) e *GnuPG*. Si segnalano i problemi più comuni che si possono incontrare quando si inviano e-mail firmate o cifrate che devono essere lette da mail client che non osservano PGP/MIME come definito nella RFC 2015 e che girano su sistemi operativi differenti. Inoltre, è presente un esempio di configurazione di procmail per inviare automaticamente le chiavi pubbliche in risposta a messaggi che le richiedano, come fanno i key server. La traduzione è stata curata da *Marco Cova*

Contents

1 Introduzione

Questo documento spiega brevemente come configurare *Mutt-i*, *PGP* nelle sue diverse versioni (2.6.x, 5.x) e *GnuPG*, così da poter usare in breve tempo un mail client in grado di gestire cifratura e firma digitale dei messaggi.

A questo scopo, saranno inclusi dei file di configurazione di esempio. Tuttavia, per ottenere la migliore performance e per sfruttare fino in fondo le possibilità dei programmi che si useranno, sarà necessario leggerne la documentazione e riconfigurare i file di esempio.

Inoltre, verranno commentati alcuni problemi derivanti dal fatto che molti mail user agent, sia per Linux che per altri sistemi operativi, non rispettano l'RFC 2015 per quello che riguarda PGP/MIME.

Sarà presentato un ulteriore esempio di configurazione di procmail, che spiegherà come fare in modo che il mail client invii una chiave pubblica su richiesta.

Questo documento è stato tradotto dall'originale spagnolo da Andrés Seco *AndresSH@ctv.es* , rivisto e corretto da Jordi Mallach Pérez *jordi-sd@softhome.net* e J.Horacio M.G. *homega@ciberia.es* . E' stato terminato nell'ottobre del 1999. Vorremmo ringraziare Roland Rosenfeld *roland@spinnaker.de* , Christophe Pernod *xtof.pernod@wanadoo.fr* , Denis Alan Hainsworth *denis@cs.brandeis.edu* e Angel Carrasco *acarrasco@jet.es* per le loro correzioni e suggerimenti.

2 Copyright and discharge of responsability

Information and other contents in this document are the best of our knowledge. However, we may have make errors. So you should determine if you want to follow the instructions given in this document.

Nobody is responsible for any damage in your computers and any other loss derived from the use of the information contained herein.

THE AUTHORS AND MAINTAINERS ARE NOT RESPONSIBLE FOR ANY DAMAGE INCURRED DUE TO ACTIONS TAKEN BASED ON INFORMATION CONTAINED IN THIS DOCUMENT.

Of course, we are open to all type of suggestions and corrections on the content of this document.

3 Inviare e ricevere e-mail su Internet

Questo documento non tratta il problema dello scambio di messaggi tra una macchina locale e altri nodi (all'interno di una rete locale o in Internet). Questo scambio dovrebbe essere portato a termine da messages transfer agent (MTA) quali, ad esempio, `sendmail` *http://www.sendmail.org* , `qmail` *http://www.qmail.org* , `exim` *http://www.exim.org* , `smail` *ftp://ftp.planix.com/pub/Smail* , etc.

In questo documento, pertanto, si presuppone che questo metodo per la ricezione e l'invio dei messaggi sia già installato e correttamente funzionante. Se si è in grado di inviare un messaggio e leggere la posta con il comando `mail` dalla linea di comando del proprio computer,

```
$ mail -s <subject> <user@domain.net>
scrivi qui il testo, e terminalo con un punto sulla prossima linea
.
```

allora deve già essere installato un qualche tipo di MTA che si occupa del trasferimento del messaggio. Altrimenti, si possono ottenere le informazioni necessarie a configurarlo nelle pagine di manuale di *smail*:

```
$ man smail
```

o del MTA che è installato, e *fetchmail*:

```
$ man fetchmail
```

o in qualche altro documento, che faccia riferimento a questi programmi.

4 Configurazione di Mutt

Il prossimo file è un buon esempio per iniziare ad usare *Mutt* in modo non molto sofisticato, utilizzando il path per il file degli alias, dei messaggi inviati e di quelli posposti. Lo si può ulteriormente personalizzare leggendo le indicazioni contenute nelle pagine di manuale di *Mutt* e in `/usr/doc/mutt/` o `/usr/doc/mutt-i/`.

Semplice esempio di `~/.muttrc`:

```
set folder=~/Mail
set alias_file=.alias
set postponed=.postponed
set record=SendMessages
set signature=.signature
my_hdr From: Name Surname <Name@domain.com>
source =.alias
```

E' necessario che la directory `~/Mail` esista, cioè che esista quella che appare come un "segno di uguale" nel file di configurazione `.muttrc` (vale a dire, `=.alias` per *Mutt* significa `~/Mail/.alias`, e `=.postponed` è `~/Mail/.postponed`). In ogni modo, è possibile tenere questi file in un'altra directory purché se ne indichi il percorso completo in `~/.muttrc`, e si abbiano i permessi necessari per lavorarci.

Si dovrà anche modificare la linea `my_hdr` con il proprio nome e il proprio indirizzo di posta elettronica. Nel file `~/Mail/.signature` si può includere la signature che apparirà in tutti i messaggi inviati.

Questo file di configurazione può risultare alla fine molto grande, così è pratica comune tenere in diversi file alcuni dei comandi di configurazione. D'ora in poi, le linee di configurazione riguardanti *PGP* o *GnuPG* e le macro della tastiera che personalizzeremo saranno tenute separate. Per fare questo, basta aggiungere le seguenti linee al file `~/.muttrc`:

```
source = ~/Mail/.mutt.macros
source = ~/Mail/.gnupgp.mutt
```

e usare i file `~/Mail/.mutt.macros` e `~/Mail/.gnupgp.mutt` per raccogliere le macro della tastiera e la configurazione di *PGP* o *GnuPG*.

Per avere informazioni più approfondite e complete riguardanti l'uso e la configurazione di *Mutt* e sulle sue caratteristiche avanzate, si veda il manuale di Mutt *http://www.mutt.org* .

5 PGP e GnuPG

Prima di usare una qualsiasi delle versioni di *PGP* con *Mutt-i*, sarà necessario configurare *PGP* appropriatamente in modo tale che il file delle chiavi pubbliche (public keys ring) e quello delle chiavi private (private keys ring) esistano. Conviene provare PGP dalla linea di comando per vedere se firma e cifra correttamente.

Esistono due versioni di *PGP* per *Unix*: la `2.6.3(i)` e `5.0(i)`, che chiameremo **PGP2** e **PGP5** rispettivamente. **GnuPG** è un nuovo programma di cifratura, sviluppato recentemente, in uno stato di sviluppo avanzato, open source e gratuito, sotto molti aspetti migliore di **PGP** (si veda il GnuPG mini howto *http://www.dewinter.com/gnupg_howto*).

Si chiarirà anche il fatto che *PGP*, essendo un programma sviluppato negli Stati Uniti, è sottoposto ad alcune leggi restrittive riguardanti l'esportazione di programmi contenenti codice criptografico. Questa è la ragione dell'esistenza di una versione internazionale di quasi tutte le versioni binarie, cosa che si può notare dalla presenza della lettera `"i"` (**pgp - pgpi**).

5.1 PGP2

PGP2 genera chiavi con l'algoritmo RSA *http://www.rsa.com* e utilizza IDEA *http://www.ascom.ch* come algoritmo di cifrazione. Entrambi gli algoritmi sono proprietari e il loro uso è regolato dai rispettivi brevetti.

Per utilizzarlo correttamente, lo si deve installare e avere una directory chiamata `~/.pgp`, contenente il file di configurazione `pgp-i.conf` e i file delle chiavi pubbliche e private, `pubring.pgp` and `secring.pgp` rispettivamente.

5.2 PGP5

Le chiavi generate da *PGP5* sono di tipo **DSS**/**DH** (Digital Signature Standard / Diffie-Hellman). PGP5 usa **CAST**, **Triple-DES** e **IDEA** come algoritmi di cifratura. PGP5 funziona con dati cifrati o firmati utilizzando *RSA* (PGP2) e, per farlo, utilizza le chiavi generate da PGP2, dal momento che PGP5 non può generare quel tipo di chiavi. Al contrario, PGP2 non è in grado di utilizzare le chiavi *DSS/DH* create da PGP5. Questo comporta problemi di (in)compatibilità, dal momento che molti continuano ad usare PGP2 sotto *Unix/Linux*.

Per utilizzare PGP5 correttamente, nella directory `~/.pgp` ci devono essere il public e il private key ring (`pubring.pkr` e `secring.skr` rispettivamente) e il file di configurazione `pgp.cfg`. Nel caso in cui si abbiano installate entrambe le versioni di *PGP* (supponiamo che PGP2 sia stato installato e configurato prima di PGP5), creeremo il file di configurazione `~/.pgp/pgp.cfg` di PGP5 come un link simbolico al file `~/.pgp/pgp-i.conf`

```
~/.pgp$ ln -s pgp-i.conf pgp.cfg
```

e aggiungeremo le seguenti linee in fondo al file `~/.pgp/pgp-i.conf`:

```
PubRing = "~/.pgp/pubring.pkr"
SecRing = "~/.pgp/secring.skr"
RandSeed = "~/.pgp/randseed.bin"
```

I file con i key ring di versioni diverse possono coesistere senza alcun problema nella stessa directory.

5.3 GnuPG

GnuPG presenta le stesse funzionalità dei precedenti programmi. A differenza di *PGP*, *GnuPG* non usa algoritmi con brevetti restrittivi. *PGP* è gratuito per uso personale, ma non per uso commerciale e il suo sviluppo non è open. Si può utilizzare gratuitamente *GnuPG* anche per scopi commerciali ed è open source, come il nostro sistema operativo preferito (anche il suo sviluppo e l'implementazione viene fatta principalmente sotto *Linux*):

Le chiavi generate da *GnuPG* sono di tipo **DSA/ElGamal** (*Digital Signature Algorithm*, noto anche come *DSS*). E' del tutto compatibile con *PGP*, eccetto per l'utilizzo da parte di quest'ultimo di algoritmi brevettati come *RSA* e *IDEA*. Comunque, è possibile implementare una certa compatibilità tra loro (vedi il GnuPG mini HOWTO *http://www.dewinter.com/gnupg_howto* per quanto concerne l'interazione con PGP2 e PGP5).

6 Integrare PGP e Mutt

L'operazione da svolgere sul messaggio in uscita (firmare, cifrare o entrambe le cose) è scelta proprio prima di premere il tasto `"y"` per inviare il messaggio, dal menu raggiungibile con l'opzione `"p"`. Una volta scelta l'operazione da compiere, cambierà soltanto la linea *PGP* nell'header del messaggio mostrato a video. Finché il messaggio non viene inviato con il comando `"y"`, non verrà chiesto né di inserire la passphrase per attivare la firma del messaggio né quali chiavi pubbliche usare per la cifrature nel caso in cui il destinatario non sia stato trovato nel nostro public key ring.

NOTA: se non si inserisce correttamente la passphrase, *Mutt* sembra "bloccarsi". Ma non è così: attende solo che venga inserita nuovamente. Per farlo, si deve premere <`Invio`> e cancellare la passphrase dalla memoria con <`Ctrl`>`F`. Poi, si deve ripetere l'invio del messaggio con `"y"` e inserire di nuovo la passphrase.

In questo modo, *Mutt* userà *PGP/MIME* per inviare il messaggio e un nuovo file apparirà nella lista dei file da inviare firmati (nel caso in cui si sia scelto solo di firmare il messaggio) o cifrerà l'intero messaggio (tutte le sue parti *MIME*). Rimarranno solo due parti MIME: la prima contenente la versione PGP/MIME e la seconda il messaggio cifrato (contenente tutte le sue parti MIME) e firmato (se si è scelto di firmare il messaggio):

Nota: se per qualche motivo il mail user agent del destinatario non è in grado di usare *MIME*, si dovrà includere la firma all'interno del corpo del messaggio. Si veda la sezione riguardante *application/pgp* alle voci 7.1 (PGP5) e 7.2 (GnuPG).

Mutt proverà a verificare la firma o a decifrare automaticamente i messaggi ricevuti che utilizzano *PGP/MIME*. Si veda la sezione 8.2 (Note e trucchi per usare Procmail), in cui si spiega come cambiare automaticamente il tipo *MIME* dei messaggi ricevuti che non lo impostano correttamente.

6.1 File opzionali di configurazione

Nella sezione che segue si trovano modifiche al file di configurazione di *Mutt* necessarie per usare facilmente 6.3 (PGP2), 6.4 (PGP5), e 6.5 (GnuPG).

A questo scopo, utilizziamo un nuovo file di configurazione che abbiamo chiamato `.gnupgp.mutt` (qualsiasi altro nome va bene, purché si inserisca il nome prescelto all'interno del file di configurazione principale `~/.muttrc`).

Lo si può fare includendo il percorso completo del file `.gnupgp.mutt` in una linea alla fine del file `~/.muttrc`. La directory in cui si inseriscono questo e altri file di configurazione opzionali può essere qualsiasi, purché si abbiano i permessi corretti (in una sezione precedente tali file sono stati inclusi nella directory `~/Mail/`), ad esempio all'interno della propria home directory. Anche sul nome non ci sono vincoli. Si può creare la directory mutt.varios:

```
~$ mkdir mutt.varios
```

in cui copiare (o creare) il file di configurazione opzionale `.gnupgp.mutt` e poi includere questo file nel file `.muttrc` col comando `source`, in questo modo:

```
source ~/mutt.varios/.gnupgp.mutt
```

Ora *Mutt* considererà le variabili di configurazioni inserite in `.gnupgp.mutt` come se fossero contenute direttamente in `.muttrc`.

Questo è un buon metodo per evitare di avere un file di configurazione enorme e disordinato, e può essere utilizzato per raggruppare altre variabili di configurazione. Per esempio, se usiamo *vim* come editor in *Mutt*, possiamo indicare a `.muttrc` di utilizzare direttamente il file di configurazione di *vim*: `.vimrc`. Innanzitutto, si deve copiare `~/.vimrc` nella nostra directory contenente i file di configurazione opzionali, `~/mutt.varios/`, e dargli un altro nome (per esempio `vim.mutt`):

```
$ cd /home/user ~$ cp .vimrc mutt.varios/vim.mutt
```

Poi, si devono cambiare le opzioni di configurazione che vogliamo siano diverse quando usiamo *vim* come editor in *Mutt*, ed infine modificare `.muttrc`:

```
set editor="/usr/bin/vim -u ~/mutt.varios/vim.mutt"
```

Con quest'ultima linea stiamo impostando `Mutt` in modo da usare un editor esterno, *Vim*, con le opzioni di configurazione desiderate.

6.2 Variabili di configurazione generale

Ci sono alcune variabili che verranno usate con tutti e tre i programmi di criptografia a chiave pubblica. Queste variabili sono booleane e possono essere **set** (attivate) o **unset** (disattivate).

Nel file di configurazione (`~/.muttrc`, `~/mutt.varios/.gnupgp.mutt` o qualsiasi altro nome si preferisca), il simbolo (#) è un commento e verrà ignorato. Lo useremo per di qui in avanti per commentare ciascuna variabile:

unset pgp_autosign

se questa variabile è attivata *Mutt* chiederà di firmare tutti
i messaggi in uscita. 6.2 ((1))

unset pgp_autoencrypt

se questa variabile è attivata *Mutt* chiederà di criptare tutti
i messaggi in uscita. 6.2 ((1))

set pgp_encryptself

salva una copia criptata di tutti i messaggi inviati che vogliamo criptare
(si deve porre `set copy=yes`).

set pgp_replysign

quando si risponde ad un messaggio firmato, la risposta sarà
anch'essa firmata.

set pgp_replyencrypt

quando si risponde ad un messaggio cifrato, la risposta
sarà anch'essa cifrata.

set pgp_verify_sig=yes

si desidera verificare automaticamente i messaggi firmati ricevuti?
Certo!

set pgp_timeout=<n>

cancella la passphrase dalla memoria cache dopo <n> secondi
che la si è inserita. 6.2 ((2))

set pgp_sign_as="0xABC123D4"

che chiave si vuole usare per firmare i messaggi in uscita?
Nota: si può impostare questa variabile col proprio user id, ma
questo può generare confusione se si hanno diverse chiavi e lo stesso user id.

set pgp_strict_enc

usa "quoted-printable" quando PGP lo richiede.

unset pgp_long_ids

non usa 64 bit key id, usa 32 bit key id.

set pgp_sign_micalg=<some>

algoritmo per il controllo dell'integrità del messaggio.
<some> va scelto tra uno dei seguenti valori:

- **pgp-md5**
  se si usano chiavi RSA
- **pgp-sha1**
  se si usano chiavi DSS (DSA)
- **pgp-rmd160**

Nelle prossime tre sezioni verranno spiegate le variabili di configurazione adatte a ciascuna delle versioni di PGP. La quarta sezione spiegherà come modificare queste variabili se si usano più versioni d PGP.

(1) Dal momento che *Mutt* chiede di inserire la passphrase ogni volta che si vuole firmare un messaggio e di scegliere un destinatario quando si vuole cifrare, può essere scomodo impostare questa variabile. Solitamente la si lascia disattivata. Questo in particolar modo è vero quando si vuole cifrare i messaggi, dal momento che non si hanno le chiavi pubbliche di tutti i destinatari.

(2) A seconda del numero di messaggi che si vuole firmare o decifrare, si decide se tenere la passphrase nella memoria cache più o meno a lungo. Questa opzione permette di non inserire la passphrase ogni volta che si firma un nuovo messaggio o se ne decifra uno ricevuto. **Attenzione:** tenere la passphrase nella memoria cache non è sicuro, soprattutto in sistemi connessi alla rete.

(3) Questo è necessario solo con la chiave che si usa per firmare. Quando la chiave viene scelta dal menu, sarà *Mutt* a calcolare l'algoritmo più appropriato.

6.3 Variabili di configurazione per PGP2

Per usare PGP2 con *Mutt-i* si dovranno aggiungere le seguenti linee al file `~/mutt.varios/.gnupgp.mutt`:

```
set pgp_default_version=pgp2
set pgp_key_version=default
set pgp_receive_version=default
set pgp_send_version=default
set pgp_sign_micalg=pgp-md5
set pgp_v2=/usr/bin/pgp
set pgp_v2_pubring=~/.pgp/pubring.pgp
set pgp_v2_secring=~/.pgp/secring.pgp
```

Come già detto, i file `~/.pgp/pubring.pgp` e `secring.pgp` devono esistere. Maggiori informazioni su PGP2 si possono ottenere col comando `man pgp`.

6.4 Variabili di configurazione per PGP5

Per usare PGP5 con *Mutt-i* si dovranno aggiungere le seguenti linee al file `~/mutt.varios/.gnupgp.mutt`:

```
set pgp_default_version=pgp5
set pgp_key_version=default
set pgp_receive_version=default
set pgp_send_version=default
set pgp_sign_micalg=pgp-sha1
set pgp_v5=/usr/bin/pgp
set pgp_v5_pubring=~/.pgp/pubring.pkr
set pgp_v5_secring=~/.pgp/secring.skr
```

Anche in questo caso, i file `~/.pgp/pubring.pkr` e `secring.pkr` devono esistere. Maggiori informazioni su PGP5 si possono ottenere col comando `man pgp5`.

6.5 Variabili di configurazione per GnuPG

Per usare *GnuPG*con *Mutt-i* si dovranno aggiungere le seguenti linee al file `~/mutt.varios/.gnupgp.mutt`:

```
set pgp_default_version=gpg
set pgp_key_version=default
set pgp_receive_version=default
set pgp_send_version=default
set pgp_sign_micalg=pgp-sha1
set pgp_gpg=/usr/bin/gpg
set pgp_gpg_pubring=~/.gnupg/pubring.gpg
set pgp_gpg_secring=~/.gnupg/secring.gpg
```

Al solito, i file `~/.gnupg/pubring.gpg` e `secring.gpg` devono esistere. Maggiori informazioni su GnuPG si possono ottenere con i comandi `man gpg.gnupg`, `man gpgm`, e `man gpg`.

6.6 Variabili di configurazione miste

Se non si ha intenzione di usare solamente una versione di questi programmi, si dovranno modificare alcune delle variabili presentate in precedenza. In realtà, si dovranno solo cambiare le variabili che impostano la versione utilizzata, rimuovendo quelle ridondanti.

Ad esempio, per usare GnuPG per firmare i messaggi, tutti i comandi di *Mutt* che fanno uso di Gnu/PGP chiameranno questo programma per apporre firme, decifrare, cifrare, verificare ecc. Per questo, si dovrà impostare la variabile di configurazione `$set_pgp_default` **una sola volta**, in questo modo:

```
set pgp_default_version=gpg
```

Invece, per usare tutti e tre i programmi, il file `~/mutt.varios/.gnupgp.mutt` potrebbe essere simile a questo:

```
set pgp_default_version=gpg     # versione del programma da usare di default

set pgp_key_version=default     # chiave da usare di default
                                # in questo caso, è gnupg a definirla

set pgp_receive_version=default # il programma utilizzato per decifrare sarà quello indicato come default
set pgp_send_version=default    # di nuovo la versione definita nella prima riga (gpg)

set pgp_gpg=/usr/bin/gpg        # dove trovare l'eseguibile di GnuPG
set pgp_gpg_pubring=~/.gnupg/pubring.gpg        # il file delle chiavi pubbliche di GnuPG
set pgp_gpg_secring=~/.gnupg/secring.gpg        # il file delle chiavi segrete di GnuPG

set pgp_v2=/usr/bin/pgp         # dove trovare l'eseguibile di PGP2
set pgp_v2_pubring=~/.pgp/pubring.pgp           # file delle chiavi pubbliche di PGP2
set pgp_v2_secring=~/.pgp/secring.pgp           # file delle chiavi private di PGP2

set pgp_v5=/usr/bin/pgp         # dove trovare l'eseguibile di PGP5
set pgp_v5_pubring=~/.pgp/pubring.pkr           # file delle chiavi pubbliche di PGP5
set pgp_v5_secring=~/.pgp/secring.skr           # file delle chiavi private di PGP5
```

7 Macro interessanti per Mutt

Mutt è molto configurabile e il suo modo operativo può essere modificato in modo molto flessibile a patto che le variabili di configurazione contenute in `.muttrc` siano ben impostate.

In seguito, si trovano alcune macro utili nel caso in cui si vogliano generare messaggi firmati senza utilizzare lo standard *PGP/MIME*, in modo da poterli inviare a destinatari che non supportano questi tipi di messaggi. E' presentata anche una macro per modificare il file degli alias e ricaricarlo senza essere costretti ad uscire da *Mutt* (quest'ultima macro non è collegata a *PGP/GnuPG* ed. è mostrata solo come un esempio della potenza delle macro in *Mutt*).

E' possibile specificare quali key binding si vogliono usare con *PGP/GnuPG*. Anche qualora alcune di queste opzioni siano già state configurate, si possono cambiare o aggiungerne delle altre semplicemente modificando il file di configurazione.

7.1 Apporre la firma nel corpo del messaggio senza usare PGP/MIME con PGP5

Prima che esistesse *PGP/MIME*, la firma di un messaggio era inclusa nel corpo del messaggio. Questo è un modo molto comune di inviare messaggi firmati in molti mail user agent.

Se si ha intenzione di firmare i messaggi in questo modo, si hanno due opzioni: non modificare il tipo *MIME* del messaggio o cambiarlo in `application/pgp`.

Per implementare queste due modi di firmare in *Mutt*, si aggiungeranno le seguenti linee al file `~/mutt.varios/mutt.macros`. Si deve aver già indicato il path di questo file nel file di configurazione principale, `.muttrc` (si faccia riferimento a 6.1 (File opzionali di configurazione)):

```
macro   compose \Cp     "F/usr/bin/pgps\ny"
macro   compose S       "F/usr/bin/pgps\ny^T^Uapplication/pgp; format=text; x-action=sign\n"
```

Ora, premendo <`Ctrl`>`p` o `S` si è in grado di includere la firma nella parte del messaggio su cui si trova posizionato il cursore, prima di inviare il messaggio.

7.2 Apporre la firma nel corpo del messaggio senza usare PGP/MIME con GnuPG

Tutto come nel caso precedente solo con GnuPG. Le macro sono:

```
macro   compose \CP     "Fgpg --clearsign\ny"
macro   compose \CS     "Fgpg --clearsign\ny^T^Uapplication/pgp; format=text; x-action=sign\n"
```

7.3 Modificare il file degli alias e ricaricarlo

Inserendo questa macro nel file `~/mutt.varios/macros.mutt` si sarà in grado di modificare il file degli alias con *vi* (ma cambiando la linea si potrà usare anche un altro editor) senza dover uscire da *Mutt*, il tutto premendo <`Alt`>`a`.

```
macro   index   \ea     "!vi ~/Mail/.alias\n:source =.alias\n"
```

7.4 Altri esempi di macro

Il seguente listato è stato ottenuto da Roland Rosenfeld e mostra come cambiare il programma utilizzato da Mutt per firmare/cifrare e come firmare senza utilizzare PGP/MIME utilizzando GnuPG

```
# ~/Mail/.muttrc.macros
# file di configurazione della tastiera per Mutt-i
# copiato, modificato e tradotto dall'originale:
#
###############################################################
# The ultimative Key-Bindings for Mutt                        #
#                                                             #
# (c) 1997-1999 Roland Rosenfeld <roland@spinnaker.rhein.de>   #
#                                                             #
# $ Id: keybind,v 1.36 1999/02/20 19:36:28 roland Exp roland $ #
###############################################################
```

```
#
# Per utilizzarlo, aggiungere la seguente linea a ~/.muttrc:
# source ~/Mail/.muttrc.macros
#

# Keybinding generici
# (per tutti i menu di Mutt, escluso il pager!)
# Con le tre opzioni successive possiamo cambiare i software utilizzato di default per cifrare:

# <ESC>1 per usare GnuPG
macro   generic \e1     ":set pgp_default_version=gpg ?pgp_default_version\n"\
"Switch to GNU-PG"

# <ESC>2 per usare PGP2
macro   generic \e2     ":set pgp_default_version=pgp2 ?pgp_default_version\n"\
"Switch to PGP 2.*"

# <ESC>5 per usare PGP5
macro   generic \e5     ":set pgp_default_version=pgp5 ?pgp_default_version\n"\
"Switch to PGP 5.*"

#NOTA: attenzione all'ultimo backspace alla fine delle macro precedenti: va inserito solo se si scrive quella linea e la
successiva su righe differenti.

# index, OpMain, MENU_MAIN
# (Menu principale)
# La prossima macro funziona solo nel menu principale (quello che appare quando si
# lancia Mutt. La combinazione <CTRL>K permette di estrarre chiavi pubbliche da un messaggio
# se questo ne contiene (lo si può sapere perché c'è una lettera K nella
# riga del messaggio):

macro  pager   \Ck     ":set pipe_decode pgp_key_version=pgp2\n\e\ek:set pgp_key_version=pgp5\n\e\ek:set pgp_key_version=gpg\n\e\ek:set pgp_key_version=default nopipe_decode\n"\ "Extract PGP keys to PGP2, PGP 5, and GnuPG keyrings"


# pager, OpPager, MENU_PAGER
# (Menu del pager)
# Permette la stessa operazione della precedente con la stessa combinazione di tasti,
# ma in questo caso dal menu del pager:

macro   pager   \e1     ":set pgp_default_version=gpg ?pgp_default_version\n"\
"switch to GNUPG"

macro   pager   \e2     ":set pgp_default_version=pgp2 ?pgp_default_version\n"\
"switch to PGP 2.*"

macro   pager   \e5     ":set pgp_default_version=pgp5 ?pgp_default_version\n"\
"switch to PGP 5.*"


# compose, OpCompose+OpGerneric, MENU_COMPOSE
# (Menu di composizione)
# Le prossime operazioni sono utilizzate nel menu di composizione
# Vale a dire, dopo che hai scritto il messaggio e lo hai chiuso per inviarlo,
# proprio prima di premere il tasto "Y" che ti permette di inviarlo al MTA.
```

```
# In questo caso, creiamo un menu che appare quando si preme la "P".
# Le opzioni nel menu saranno collegate a MENU_PGP. Queste sono
# le opzioni principali (cifratura e firma):

bind    compose p       pgp-menu

# Dal momento che molti programmi non possono usare PGP/MIME (soprattutto quelli M$), la combinazione <CTRL>P
# ci permetterà di firmare i messaggi alla vecchia maniera (Application/PGP):

macro   compose \CP     "Fgpg --clearsign\ny"

# La seguente, <CTRL>S, ci permetterà di firmare usando PGP/MIME con la chiave privata
# che abbiamo impostato di default. Questa macro non è necesaria, dal momento che
# possiamo ottenere lo stesso risultato dal menu "P":
macro   compose \CS     "Fgpg --clearsign\ny^T^Uapplication/pgp; format=text; x-action=sign\n"
```

Si possono aggiungere altre macro e alcune sono già configurate e attivate di default in versioni più recenti di Mutt. Alcune altre opzioni includono

- <CTRL>K (estrae chiavi pubbliche da un messaggio)
- <ESC>K (allega una chiave pubblica a un messaggio)
- <CTRL>F (quando si usa la passphrase per firmare o decifrare un messaggio, questa rimane in memoria. Con questo comando la si può cancellare dalla memoria)
- ecc...

Per vedere quale altre opzioni sono attive, si deve accedere al menu di aiuto (?)

8 Note e trucchi per usare Procmail

8.1 Configurare Procmail per inviare automaticamente le proprie chiavi pubbliche

Dato che questo non è lo scopo principale di questo Howto, aggiungeremo che il modo più sicuro per ottenere una chiave pubblica da qualcuno è che questi ce la dia di persona, a mano.

Poiché molte volte questo non è facilmente realizzabile, le persone si scambiano chiavi pubbliche via posta elettronica o cercandole in un key server, per quanto nessuno di questo metodi assicuri che la chiave ricevuta sia realmente la chiave di chi dovrebbe essere. Si possono usare altri mezzi di comunicazione considerati "sicuri" (cercare il possessore della chiave nell'elenco telefonico e chiedendogli di leggere la sua "fingerprint" per confrontarla con quella della chiave ottenuta attraverso percorsi non sicuri):

Quello che vedremo è un "trucco" da inserire nel file `.procmailrc` per restituire la propria chiave pubblica al mittente di un messaggio che abbia un ben determinato `Subject`:

```
:0 h
* ^Subject:[    ]+\/(|send)[    ]+key pub\>.*
| mutt -s "Re: $MATCH" `formail -rtzxTo:` </clau/mykey.asc
```

Ciò che è indicato nel precedente paragrafo è: abbiamo una copia in ASCII della nostra chiave pubblica, in una certa directory (in questo caso in `/clau`) in un file chiamato `mykey.asc`. Quando procmail riceve un messaggio che include "send key pub" nel `Subject:`, invia quel file al mittente.

IMPORTANTE: tra le parentesi è contenuto **uno spazio** e **una tabulazione**.

8.2 Verificare e decifrare automaticamente un messaggio senza PGP/MIME

Quando si riceve un messaggio firmato che utilizza PGP/MIME e lo si apre col proprio MUA preferito (Mutt, no?), questo riconosce il messaggio come PGP/MIME e controlla la firma se si possiede la chiave del mittente. Questi messaggi sono quelli con la lettera "S" nella prima parte della riga del messaggio in Mutt:

```
36  S  05/09 Andres Seco Her ( 12K) Al fin
```

Invece i messaggi cifrati hanno la "P":

```
12  P  03/24 Andres Seco Her (6,3K) Re: FW: Re: Mutt - pgp/gnupg
```

Ma se il messaggio è cifrato e ha il tipo MIME "application/pgp", quando lo si apre Mutt non ne controlla la firma e la firma è contenuta all'interno del corpo del messaggio, come in questo caso:

```
-----BEGIN PGP SIGNED MESSAGE-----

Date: Tue, 25 May 1999 13:04:26 +0200
From: La Corporación <bill@reboot.com>
Subject: Actualización S.O.
To: Sufrido Usuario <pepe@casa.es>


Sufrido usuario:

le comunicamos que puede usted adquirir la última actualización del
programa O.E. con la adquisición de nuestro sistema operativo reboot99
por el módico precio de ... etc.


-----BEGIN PGP SIGNATURE-----
Version: 2.6.3ia
Charset: noconv

iKBGNpUBX0235VapRBUy1KklAQGl9wQA3SBMioObbbajHAnyKMOlx3tcgNG7/UVC
AbqXcUnyGGOo13Nbas95G34Fee3wsXIFo1obEfgiRzqPzZPLWoZdAnyTlZyTwCHe
6ifVpLTuaXvcn9/76rXoI6u9svN2cqHCgHuNASKHaK9034uq81PSdW4QdGLgLoeB
vnGmxE+tGg32=
=Xidf
-----END PGP SIGNATURE-----
```

Per verificarlo, lo si deve salvare e usare la linea di comando. Tuttavia, è possibile convertire il tipo MIME di questo messaggio con *Procmail* per permettere a *Mutt* di riconoscerlo come *PGP/MIME*. Si deve solo aggiungere a `.procmailrc`:

```
:0
* !^Content-Type: message/
* !^Content-Type: multipart/
* !^Content-Type: application/pgp
{
```

```
        :0 fBw
        * ^-----BEGIN PGP MESSAGE-----
        * ^-----END PGP MESSAGE-----
        | formail \
            -i "Content-Type: application/pgp; format=text; x-action=encrypt"

        :0 fBw
        * ^-----BEGIN PGP SIGNED MESSAGE-----
        * ^-----BEGIN PGP SIGNATURE-----
        * ^-----END PGP SIGNATURE-----
        | formail \
            -i "Content-Type: application/pgp; format=text; x-action=sign"
    }
```

Si può notare che questo è valido per messaggi firmati e cifrati di tipo application/pgp.

8.3 Cambiare il tipo MIME per messaggi contenenti chiavi senza PGP/MIME

Se si riceve un public key block da un MUA che non rispetta *PGP/MIME*, si deve salvare il corpo del messaggio e poi inserirlo nel proprio public key ring. Però, inserendo queste linee in `.procmailrc`, si potrà includere la chiave direttamente da mutt.

```
:0 fBw
* ^-----BEGIN PGP PUBLIC KEY BLOCK-----
* ^-----END PGP PUBLIC KEY BLOCK-----
| formail -i "Content-Type: application/pgp-keys; format=text;"
```

Grazie a Denis Alan per questa nota su procmail.

9 Scambiare messaggi firmati/cifrati con MUA e piattaforme differenti

All'inizio, la firma era inserita all'interno del testo. In seguito, è stato introdotto il tipo MIME `application/pgp` ad indicare che il successivo attachment era la firma o il messaggio cifrato. Infine, con le specifiche PGP/MIME, è stato possibile isolare la firma dal messaggio originale, così da non modificarlo assolutamente e in modo tale che chi non avesse PGP potesse vedere il messaggio come era originariamente (nel caso di messaggio firmato), senza alcun testo aggiunto all'inizio o alla fine da PGP.

Attualmente, solo pochi mail user agent (MUA) sono in grado di integrare PGP e usare lo standard PGP/MIME. Così è necessario inviare i messaggi utilizzando il vecchio metodo quando si sa che il destinatario non supporta PGP/MIME.

In Linux, i mail user agent che riconoscono PGP/MIME sono mutt-i e pine. In Windows, solo Eudora nelle versioni 3.x e 4.x può usare PGP/MIME. Se conoscete altri mail user agent che supportano PGP/MIME, ditemelo via e-mail, e li includerò qui.

10 Programmi e versioni utilizzati

Per scrivere questo documento abbiamo utilizzato le seguenti versioni di Mutt:

- Mutt 0.93i - non si può usare GnuPG con questa versione.
- Mutt 0.95.3i - possono essere usate tutte le versioni di PGP e GnuPG

Di seguito, le versioni di PGP e GnuPG:

- PGPi 5.0
- GnuPG 0.4.3
- GnuPG 0.9.4

11 Ulteriori informazioni

La documentazione originale da cui questo documento è stato tratto può essere trovata nelle pagine di manuale di "mutt", "pgp", "pgp5", "gnupg", "procmail", nelle rispettive directory sotto /usr/doc e nei siti:

- Home Page Ufficiale di Mutt- *http://www.mutt.org*
- Sito Principale di GnuPG - *http://www.gnupg.org*
- Sito di PGP International - *http://www.pgpi.com*
- Home Page Ufficiale di Procmail - *http://www.procmail.org*

Le RFC cui si fa riferimento in questo documento sono:

- 1847 - Security Multiparts for MIME: Multipart/signed and Multipart/encripted
- 1848 - MIME Object Security Services
- 1991 - PGP Message Exchange Formats
- 2015 - MIME Security with Pretty Good Privacy (PGP)
- 2440 - OpenPGP Message Format

e possono essere trovate in /usr/doc/doc-rfc e in vari siti su Internet, ad esempio *http://metalab.unc.edu* e *http://nic.mil* . Si possono ottenere informazioni sulle RFC in *RFC-INFO@ISI.EDU*