OpenBSD su un netbook

Antonio Frecentese

7-4-2024

Introduzione

Ho sempre cercato delle alternative a Windows, con i suoi *crash* di sistema e dei programmi, la sua sempre maggiore pesantezza in termini di risorse e la sua invadenza in termini di *privacy*. Più volte ho provato Linux (RedHat, Debian, Salix, Crunchbang, Manjaro), ma anche NetBSD e PCBSD.

L'ultima mia prova è OpenBSD. Dopo un esperimento su una macchina virtuale, ho deciso di acquistare un piccolo *netbook* e dedicarlo per intero a questo sistema operativo. Dal momento che intendo usare tale PC quasi solo come macchina da scrivere, ho scelto di limitarmi a usare i programmi presenti nell'installazione di base o comunque il più possibile leggeri.

Tenete presente che quando troverete scritto ad esempio `^g` intenderò dire "tenete premuto il tasto CTRL e premete `g`".

Installazione

1. Preparazione

I *netbook* che ho usato non hanno un lettore CD, quindi ho scaricato dal sito di OpenBSD la ISO e l'ho installata in una scheda SD. Acceso il computer, bisogna accedere al BIOS per poter far avviare l'installazione dalla scheda SD invece che dal disco rigido (premere poi `F10` per salvare e far ripartire).

1.1. Asus eeePC 1001 PXD

Si accede al BIOS con `F2`.

1.2. HP Mini

Si accede al BIOS con `F10`; qui devo anche togliere una funzione preinstallata che consente di avviare subito un menu semplificato per accedere alla posta elettronica o a Internet o a Windows.

Nel caso dell'HP Mini, l'installazione non si avvia. Come suggerito da un gruppo di utenti in una *chat* Matrix, avvio il sistema indicando `bsd -c`, poi indico `disable acpipci [INVIO] exit [INVIO]`. A questo punto il *boot* prosegue senza intoppi. Dopo sistemerò questa impostazione in modo da non doverla fare a ogni avvio.

2. Installazione

Premo il tasto `I` per dare inizio all'installazione, scelgo la tastiera `it`, poi lo *hostname* e quale interfaccia usare per

la connessione a Internet. In entrambi i PC mi viene restituito errore, ma non importa, sistemerò dopo. Inserisco il nome di dominio DNS e il *name server* (`none`).

Ora è il momento di scegliere la *password* di *root*. Alla domanda se far partire il servizio `sshd` di *default* rispondo `yes`, e lo stesso alla domanda se `X` deve esser lanciato tramite `xenodm`.

A questo punto, l'utente normale: scelgo nome utente e *password*, e alla domanda "Allow root ssh login?" rispondo `no`.

Comincia ora la parte per me più complessa dell'installazione. Come *root disk* indico `sd0` e scelgo poi di dedicare l'intero disco rigido a OpenBSD `e`ditando lo schema proposto: cancello tutte le partizioni proposte, ne `a`ggiungo una da dedicare a OpenBSD grande il 95% dello spazio disponibile e poi una seconda da usare come *swap* per la restante parte del disco rigido.

Come posizione dei *set* da installare indico `disk`, dato che si trovano sulla scheda SD; rispondo `no` alla domanda se si tratti di una partizione già montata e `sd1` (ovvero la scheda SD) a quella che chiede dove si trovino i *set*. A questo punto scelgo di installare tutti i *set*. Imposto il fuso orario (`Europe/Rome`) e do finalmente il comando `reboot`. Tolgo la scheda SD e mi godo il sistema appena installato.

3. Connessione a Internet

3.1. ASUS eeePC 1001 PXD

```
# vi /etc/hostname.athn0
```

Qui scrivo

`nwid` *nome_della_rete* `wpakey` *password_della_rete*
`dhcp`

Fatto questo, per connettermi a Internet (adesso e in caso di disconnessione) basta lanciare:

```
# sh /etc/netstart athn0
```

3.2. HP Mini

Con questo PC, c'è un problema: vedo disponibile solo `re0`, che è la scheda Ethernet. Eppure so per certo che la scheda wifi c'è.

```
# dmesg | grep "not configured"
```

mi rivela che non è stata configurata la scheda Broadcom BCM4315. Purtroppo, ricercando in Internet, scopro che il *chip* di questa scheda non è supportato da OpenBSD, quindi per il momento dovrò dunque accontentarmi della connessione Ethernet: collego allora il cavo al computer e al *router* e poi do i comandi

```
# echo "dhcp0net6 autoconfig" > /etc/hostname.re0
# sh /etc/netstart re0
```

Dopo alcune ricerche su Internet, ho comprato per pochi euro un adattatore TP-Link TL-WN725N. È tanto piccolo che una volta inserito in una porta USB posso lasciarlo lì anche quando ripongo il computer nella borsa. `ifconfig` mi dice che viene riconosciuto dall'HP Mini come `urtwn0`, quindi modifico `/etc/hostname.urtwn0` inserendo

```
join mia_rete wpakey mia_password
join "mia rete 2" wpakey mia_password_2
dhcp
```

Poi lancio `sh /etc/netstart urtwn0`, *et voilà!* La connessione wifi è ora attiva. Lenta, dato che questo adattatore non è un gran che, ma sufficiente per i miei scopi. Notare che ho indicato due connessioni, l'una di seguito all'altra, di modo tale che se la prima non è disponibile venga usata la seconda, e che nella seconda istruzione ho messo il nome della connessione tra virgolette perché tale nome contiene degli

spazi.

4. doas

Imposto poi doas, che serve per poter utilizzare comandi propri dell'utente *root* senza dover accedere al sistema come *root* stesso e con la *password* del normale utente.

Alcuni comandi per me fondamentali sono mount e umount, quindi creo quindi (come *root*) il *file* /etc/doas.conf scrivendovi:

```
permit :wheel as root
permit nopass root
permit nopass :wheel as root cmd mount
permit nopass :wheel as root cmd umount
```

Ora posso, per esempio, inserire una chiavetta USB e montarla con doas mount /dev/sd1i ./pen (ovviamente dopo aver creato questa *directory*) senza bisogno di inserire la *password*.

Dalla versione 7.4 di OpenBSD, è possibile aggiungere un utente al gruppo _shutdown per fare in modo che possa spegnere il PC o riavviarlo senza bisogno della *password*:

```
# usermod -G wheel nome_utente
# usermod -G _shutdown nome_utente
```

Si può controllare la presenza nel *file* /etc/group. Adesso è sufficiente che l'utente faccia di nuovo *login* per rendere operativa la modifica.

5. Qualche programma

Installo subito alcuni pacchetti per me utili:

```
$ doas pkg_add w3m zathura zathura-ps zathura-pdf-mupdf heirloom-doctools nvi vifm nsxiv liberation-fonts scheherazade leafpad ntfs_3g conky wget xdg-utils
```

6. Ambiente grafico: `cwm`

Mi piace molto il *window manager* `cwm`, già contenuto nell'installazione di base.

`cwm` non ha una sua barra di stato, che trovo comunque utile. Io utilizzo `conky` personalizzandolo appena per fargli svolgere questa funzione. `.xsession` conterrà allora:

```
xsetroot -solid black
conky &
cwm
```

La prima riga imposta uno sfondo nero (non mi interessa avere un'immagine come sfondo), la seconda lancia `conky`, la terza il *window manager*.

Ecco il contenuto attuale del mio *file* `~/.conkyrc`:

```
default_color green
double_buffer yes
own_window yes
own_window_type override
alignment top_middle
draw_borders no
update_interval 1
gap_x 0
gap_y 5
use_spacer left
TEXT
Desktop: ${exec xprop -root _NET_CURRENT_DESKTOP |cut -f2 -d=tr -d "[[ ]]"} | ${time %a} ${time %v} | \
Battery: $apm_battery_life $apm_battery_time $apm_adapter
```

Anche `~/.cwmrc` va personalizzato per poter usare bene `conky` e per avere delle combinazioni di tasti a me congeniali per le normali funzioni. Ricordate che `C` indica il tasto CTRL, `M` il tasto ALT, `S` il tasto SHIFT e `4` il tasto "super", ovvero il tasto con il logo di Windows. Ogni tanto cambio qualcosa, ma

ecco alcune righe del mio *file*:

```
unbind-key all
bind-key 4-Return xterm
bind-key S4-r restart # reloads configuration
bind-key S4-q quit # logs out as user
bind-key 4-e menu-cmd # application search menu
bind-key 4-m window-maximize
bind-key S4-x window-delete
bind-key 4-h window-htile
bind-key 4-v window-vtile
bind-key M-Tab window-cycle
bind-key 4-1 group-only-1
bind-key 4-2 group-only-2
bind-key S4-1 window-movetogroup-1
bind-key S4-2 window-movetogroup-2
# default, but just to remember
# bind-mouse M-1 window-move
# bind-mouse M-2 window-resize
command xterm xterm
command w3m "xterm -e w3m http://duckduckgo.com"
command irssi "xterm -e irssi"
command aerc "xterm -e aerc"
command leafpad leafpad
command zathura zathura
command vifm "xterm -e vifm"
# sets window border in pixels
borderwidth 3
color activeborder red
color inactiveborder gray
# minimum distance to snap-to adiacent edge, in
pixels
snapdist 16
# sticky = new windows assigned to the current
group
sticky yes
# leaves a gap at the top so that windows won’t
```

```
cover conky
 gap 20 0 0 0
```

7. Supporto per la lingua giapponese

Dal momento che nei miei testi userò anche la lingua giapponese, non basta avere un *editor* che accetti la codifica UTF-8: bisogna anche dotare il sistema di un metodo per inserire i caratteri che mi servono, il cosiddetto IME (*Input Method Editor*). Anni fa usavo `kinput2` in accoppiata con `canna-server`, ora invece uso `uim` e `anthy`:

```
 $ doas pkg_add uim uim-gtk anthy ja-sazanami-ttf
ja-kanjistrokeorders-ttf
```

Ora bisogna impostare le variabili d'ambiente corrette e lanciare il servizio apposito, perciò inserisco nel *file* `~/.profile` le seguenti righe (attenzione: le righe che qui di seguito cominciano e finiscono con un * vanno in realtà inserite come un'unica riga, senza spazi e senza asterischi):

```
 *export TROFFONTS=/usr/local/share/fonts/sazanami:*
 */usr/local/share/fonts/freefont:*
 */usr/local/share/fonts/Liberation:*
 */usr/local/share/fonts/scheherazade:*
 */usr/local/share/fonts/KanjiStrokeOrders*
 export XMODIFIERS=´@im=uim´
 export GTK_IM_MODULE=´uim´
 export QT_IM_MODULE=´uim´
 export LANG=it_IT.UTF-8
```

Allo stesso modo apporto tali modifiche all'inizio di `~/.xsession`, nel quale, subito prima di lanciare `cwm` faccio partire il servizio (attenzione: le righe che qui di seguito cominciano e finiscono con un * vanno in realtà inserite come un'unica riga, senza spazi e senza asterischi):

```
 *xset +fp /usr/local/share/fonts/freefont,*
 */usr/local/share/fonts/sazanami,*
 */usr/local/share/fonts/KanjiStrokeOrders*
```

```
export XMODIFIERS=´@im=uim´
export GTK_IM_MODULE=´uim´
export QT_IM_MODULE=´uim´
export LANG=it_IT.UTF-8
uim-xim &
uim-toolbar-gtk &
cwm
```

A questo punto lancio `uim-pref-gtk` e deseleziono i sistemi di *input* che non mi interessano, lasciando solo `Anthy (UTF-8)`, `m17n-ja-anthy`, `m17n-ar-translit` (per l'arabo mediante traslitterazione) e `m17n-zh-tonepy` (per il cinese; bisogna mettere il *pinyin* e aggiungere dopo la sillaba il tono per poter avere la scelta dei caratteri con tale traslitterazione), per poi impostare `uim` in modo da poter cambiare tra i vari sistemi premendo CTRL-SHIFT e da attivare/disattivare il sistema attualmente in uso con SHIFT-SPACE. Faccio anche in modo che la finestra di `uim` sia sempre visibile, per poter avere sempre sott'occhio una *mini-toolbar* che mi mostri quale sistema io stia usando in quel momento.

Occorre poi impostare un *font* per la finestra di *pre-input*, in modo che, usando `uim-xim`, i caratteri che si inseriscono vengano visualizzati correttamente, prima di premere SPAZIO per convertirli in *kanji*. Per questo motivo nel *file* `~/.Xdefaults` scrivo

```
XTerm*ximFont: -misc-fixed-medium-r-normal--14-130-75-75-c-140-jisx0208.1983
```

Ora è sufficiente riavviare la sessione e rientrare per rendere effettivo il cambiamento.

8. Sviluppo web

Mi diletto anche nello sviluppo di siti *web*, e fortunatamente il *server web* viene già installato nel sistema

base.

Nel *file* `/etc/httpd.conf` scrivo

```
server "www.frech.altervista.org" {
listen on * port 80
root "/htdocs/www.frech.altervista.org"
directory index index.htm
}
types {
include "/usr/share/misc/mime.types"
}
```

Poi eseguo i comandi

```
# mkdir -p /var/www/htdocs/www.frech.altervista.org
# chown antonio /var/www/htdocs/www.frech.altervista.org
```

Infine metto in questa cartella i *files* del mio sito e lancio i comandi

```
# rcctl enable httpd
# rcctl start httpd
```

Ora posso aprire il *browser* e andare su `http://localhost/` per aprire il mio sito, e ad ogni successivo riavvio del PC non dovrò rilanciare il servizio.

9. Altre personalizzazioni

Non mi piace avere in basso a destra la finestra di `xconsole` che mostra tutto ciò che accade: ruba spazio prezioso in uno schermo già piccolo di suo e non ho quasi mai motivo di guardare ciò che riporta. Per disattivarla entro dunque in `/etc/X11/xenodm/Xsetup_0` e commento la linea che comincia con `xconsole`. Già che ci sono, nello stesso *file* inserisco `xsetroot -solid black` in modo tale che in fase di *login* lo sfondo sia nero invece che grigio a punti.

Nel *file* `/etc/X11/xenodm/Xresources` aggiungo la riga `xlogin.Login.echoPasswd: true` in modo tale da far apparire degli asterischi mentre inserisco la *password* per accedere al sistema.

Voglio impostare il carattere della finestra di terminale e inoltre voglio scritte bianche su fondo nero, quindi nel *file* `~/.Xdefaults` scrivo

```
XTerm*faceName: Liberation:size=8
XTerm*foreground: white
XTerm*background: black
```

Poi, con `xrdb -merge .Xdefaults` rendo effettivo il cambiamento senza bisogno di riavviare.

Inoltre, voglio cambiare il *prompt* in modo che mostri la *directory* corrente (per intero), quindi in `~/.profile` inserisco

```
PS1='${PWD}'
```

Per `nvi`, invece, nel *file* `~/.exrc` inserisco le righe:

```
:set nu
:set ic
:set smd
:set ruler
:set report=1
:set verbose
```

Nel *file* `/etc/fstab` aggiungo `noatime` nei parametri per aprire le varie partizioni (esclusa quella di *swap*), in modo da rendere leggermente più rapide le operazioni di lettura e scrittura, trattandosi di *netbook* poco performanti.

10. Aggiornamenti

Con `pkg_delete -a` si possono rimuovere le dipendenze non più usate di un pacchetto che si è rimosso, e con `pkg_add -u` si aggiornano i pacchetti installati nel proprio sistema.

A volte capita che uno dei *mirror* non funzioni e bisogna quindi cambiarlo: basta modificare il *file* `/etc/installurl` cambiando l'indirizzo con quello di un *mirror* differente.

Per aggiornare il sistema a una nuova versione si usa il comando `sysupgrade`, seguito poi da `pkg_add -u` per aggiornare i programmi e da `syspatch` per correggere i problemi di sicurezza rilevati dagli sviluppatori.

10.1. HP Mini

Io ho avuto problemi nell'eseguire `sysupgrade` con l'HP Mini, nel senso che dopo aver scaricato ciò che serve per l'aggiornamento, il programma si blocca dopo aver fatto il `mount` del *filesystem*.

Per questo motivo devo procedere col metodo manuale: scaricare da uno dei *mirror* il *file* `bsd.rd`, copiarlo nella *directory* `/`, riavviare il sistema e al *boot* indicare `boot bsd.rd -c`, poi ancora `disable acpipci [INVIO] exit [INVIO]` e finalmente far ripartire il sistema... il quale si bloccherà ancora nello stesso punto di prima. Nessun problema: `^c` per uscire dallo *script*, eseguiamo `upgrade`, e ora è tutto come deve essere. Si seleziona la tastiera italiana, il disco di *root*, come origine dei pacchetti `http` e poi specificare un *mirror* vicino, in modo da rendere le operazioni più veloci.

Tutto prosegue come per la prima installazione, finché alla fine si fa ripartire il sistema. Tuttavia, visto che il *kernel* è cambiato in seguito all'aggiornamento, bisogna ricordarsi di reimpostare il parametro per disabilitare l'ACPI. Quindi al *boot* si dà il comando `boot -c` e poi ancora `disable acpipci [INVIO] exit [INVIO]` e il sistema ora riparte senza intoppi con la nuova versione.

Si accede poi come *root* e da terminale si digita:

```
config -e -f /bsd
disable acpipci
quit
```

Fatto. Al prossimo riavvio, non dovrò più preoccuparmi di questa impostazione. Almeno fino al prossimo aggiornamento, ovviamente...

Ricordiamoci infine di aggiornare i programmi col comando `pkg_add -u` e di eseguire `syspatch`.

11. Ripristino scheda SD

Giusto per non avere problemi in futuro, riporto qui come risolvere un piccolo problemino riscontrato formattando la scheda SD usata per l'installazione per poterla poi usare per trasportare dati: Windows (ebbene sì, lo uso ancora) non riconosce più la sua effettiva capienza.

Inserisco la scheda e lancio da terminale il programma `diskpart.exe`. Ecco ora i comandi da eseguire:

1) `list disk` per mostrare l'elenco dei dischi rigidi che Windows vede attualmente. Normalmente il disco 0 è quello su cui è installato il sistema; la mia scheda ha il numero 2 perché ho anche un altro HD collegato. Posso comunque identificare bene la scheda, perché può contenere poco meno di 2 GB, mentre gli altri HD sono ben più capienti
2) `select disk` *n*, dove al posto di *n* devo mettere il numero del disco corrispondente alla scheda SD da formattare
3) `clean` per pulire la scheda
4) `create partition primary`
5) `exit`

Ora si può scollegare e poi collegare di nuovo la scheda al PC con Windows e formattarla; io uso FAT come sistema e

seleziono “*default allocation size*”. Poi lancio la formattazione. *Voilà!*

12. Ripristino password di *root*

Nel malaugurato caso in cui non ci si ricordi la *password* di *root*, occorre procedere in questo modo. Innanzitutto, non appena acceso il PC e comparso il *prompt* bisogna indicare `boot -s` per entrare in modalità utente singolo. Poi bisogna effettuare il `mount` di due partizioni in modalità *read-write*:

```
# fsck -p / && mount -uw /
# fsck -p /usr && mount -uw /usr
```

Successivamente occorre eseguire `passwd` per impostare appunto la *password* dell’utente *root*; dato che si è in modalità utente singolo, si avranno già i privilegi di *root* e quindi non verrà chiesta la *password*.

Per finire occorre far ripartire il sistema, o col comando `reboot` oppure premendo `^D` per proseguire col normale procedimento di *boot*.

Utilizzo

Bene, ora che le personalizzazioni sono terminate, posso cominciare a usare il PC. Da questo momento in poi si tratta solo di rendere questo piccolo PC una sorta di macchina da scrivere per i miei racconti, le mie ricerche, i miei libri, con qualche piccola divagazione ogni tanto.

Userò quindi programmi il più possibile leggeri e che richiedono poche risorse, come `nvi` per scrivere i testi (sì, lo so che il sistema base contiene già `vi`, ma questo non ha il supporto per la codifica Unicode), `troff` per creare i miei documenti in formato PDF (al posto del mastodontico LaTeX), `tmux` per gestire meglio le mie finestre di terminale, `irssi` per cercare aiuto in Rete, e così via.

Nei prossimi capitoli vedremo quindi come usare questi programmi. Attenzione: si tratterà di informazioni date in forma di prontuario, come guida veloce di riferimento. Per ogni spiegazione più approfondita, meglio studiare le pagine `man` o le guide che si possono trovare su Internet. Ma prima di tutto, vediamo prima delle cose di base.

1. `xterm`

Premendo ^*tasto_sinistro* si apre un menu di opzioni del programma; con ^*tasto_di_mezzo* un differente menu di opzioni; con ^*tasto_destro* un ulteriore menu dal quale scegliere la grandezza dei *font* con cui visualizzare il testo.

Si può selezionare del testo in una finestra di terminale tenendo premuto il tasto sinistro del *mouse* per evidenziare

il testo stesso e poi premere il tasto centrale del *mouse* per incollarlo in un'altra finestra di terminale o in un programma come `leafpad`.

2. Trovare *files*

- `$ find . -type f -iname miofile -group www -mtime 10` trova un *file* chiamato `miofile` da qualche parte nella *dirrectory* attuale, modificato negli ultimi 10 giorni e appartenente al gruppo `www`, senza fare distinzione tra maiuscole e minuscole
- `$ find` */un/certo/percorso/* `-type d -name MiaDIR` ci aiuta a trovare una *directory* di nome `MiaDIR`
- `$ find . -type f -iname *.pdf -size +10M -size -100M` trova invece, da qualche parte nella *directory* corrente e senza fare distinzione tra maiuscole e minuscole, un *file* in formato PDF che sia più grande di 10 MB ma più piccolo di 100 MB.

3. Gestire il disco

- `$ df -h` mostra l'utilizzo totale del disco rigido
- `$ du -hs` *mia_directory* mostra lo spazio utilizzato occupato da quella *directory*
- `$ du -ha` *mia_directory* mostra invece recursivamente lo spazio occupato da tutti i *files* contenuti in quella *directory*

4. Gestire i permessi

Se si esegue il comando `ls -l` *mia_dir*, si vedono in maniera ricorsiva tutti i nomi dei *files* contenuti in quella data cartella, con alcune interessanti informazioni:

- il primo carattere indica se si tratta di un *file* normale (`-`), una *directory* (`d`) oppure un *link* (`i`)

- i successivi nove caratteri, presi a gruppi di tre, indicano i permessi relativi a quel dato *file*; vedremo dopo di cosa si tratta
- compare poi una cifra che indica il livello delle sotto-*directory*
- compaiono poi il nome dell'utente creatore del *file*, che quindi ne è il proprietario, e poi il nome del gruppo a cui il *file* appartiene
- infine abbiamo le dimensioni, la data dell'ultima modifica e il nome appunto del *file*

Ora, i permessi indicano ciò che può fare il proprietario del *file*, ciò che può farne un membro del gruppo cui appartiene e cosa può fare qualunque altro utente. I tre permessi sono lettura, scrittura ed esecuzione, indicati rispettivamente dalle lettere `r`, `w` e `x`.

Se si tratta di un ***file***:

- per "lettura" si intende che il *file* può essere letto, cioè si può leggere il suo contenuto;
- per "scrittura" si intende che il suo contenuto può essere modificato o rimosso;
- per "esecuzione" si intende che il *file* può essere eseguito come un programma.

Se si tratta di una ***directory***:

- per "lettura" si intende che si può leggere il suo contenuto, quindi vedere i nome dei *files* contenuti al suo interno;
- per "scrittura" si intende che si può aggiungere o rimuovere dei *files* al suo interno;
- per "esecuzione" si intende che un utente possa entrarvi.

Questo significa per esempio che se si ha `rwxr-xr-x` allora il proprietario del *file* può leggerlo, modificarlo o eseguirlo (`rwx`), mentre altri membri del gruppo e gli altri utenti possono solamente leggerlo ed eseguirlo ma non

modificarlo (`r-x`). Un altro utente potrebbe comunque leggere il *file*, farne una copia per esserne il proprietario e poi modificarla.

È possibile modificare queste impostazioni col comando `chmod`. Ecco alcune situazioni:

- si potrebbe voler usare `chmod ug+r` *file* per poter dare all'utente (`u`, cioè il proprietario) e al suo gruppo (`g`) il permesso di lettura (`r`)
- si può anche togliere il permesso di modifica `w` e di esecuzione `x` a ogni altro utente: `chmod o-wx` *file*

Questi permessi possono essere impostati anche in un'altra maniera, coi numeri ottali, cioè con una sequenza di tre cifre che indicano rispettivamente i permessi per il proprietario del *file*, per il gruppo a cui il *file* appartiene e per gli altri utenti. Ma quale cifra?

Ottale	Sigla	Permesso
0	`---`	Nessun permesso
1	`--x`	Esecuzione
2	`-w-`	Scrittura
3	`-wx`	Scrittura ed esecuzione
4	`r--`	Lettura
5	`r-x`	Lettura ed esecuzione
6	`rw-`	Lettura e scrittura
7	`rwx`	Tutti i permessi

Quindi per esempio `chmod 750` *mio_file* indica che il proprietario di *mio_file* potrà leggerelo, modificarlo ed eseguirlo, i membri del gruppo a cui appartiene *mio_file* lo potranno solo leggere ed eseguire e gli altri utenti non potranno farci nulla.

Ricordiamo poi il comando `chown` *nuovo_utente nome_file* per cambiare il proprietario di *nome_file* e il

comando `chgrp` *nuovo_gruppo nome_file* per assegnarvi un nuovo gruppo.

5. Date e scadenze

Col comando `cal` viene visualizzato nella finestra del terminale un semplice calendario del mese in corso. Ma se occorre invece impostare dei promemoria, si usa `calendar`. Questo programma legge un *file* di nome `./calendar` oppure `./calendar/calendar` scritto in questo modo:

```
16/12/2023 Andare dal dentista
Dec 17     Prove di canto
```

Si ha insomma la data (scritta in uno di questi due formati), un carattere di TAB (non spazio) e poi il testo del promemoria, scritto tutto su una sola riga. Ora, eseguendo `calendar`, viene viene visualizzato l'appuntamento del giorno in corso e del giorno seguente. Naturalmente, se si inserisce questo comando in `.profile`, diventa utile poiché il promemoria compare non appena si accede al sistema. L'unica cosa forse tediosa è dover editare ogni volta il *file* `calendar`, ma se non altro è una soluzione semplice.

File browser testuale: `vifm`

`vifm` usa due pannelli per mostrare i *files* (un po' come Midnight Commander). È possibile passare alla modalità con un solo pannello premendo `^wo` (per poi tornare alla normale visualizzazione a due pannelli con `^wv`).

Per selezionare o deselezionare un *file* si preme `t`. Con `za` si può visualizzare anche i *files* nascosti, e con lo stesso comando li si rinasconde. Con `dd` si cancella il *file* selezionato, con `cw` lo si rinomina. Per copiare un *file* nella *directory* visualizzata nell'altro pannello si preme `yy` (per copiarlo in memoria) e poi `p` per incollarlo. Per passare da un pannello all'altro, invece, si preme TAB oppure SPAZIO.

Per aprire un *file* vi si preme sopra INVIO oppure la freccia a destra. Per aprire un'immagine viene lanciato `sxiv`.

Il programma va già bene così, ma per adattarlo ai miei scopi ho modificato il *file* `./config/vifm/vifmrc` affinché come *editor* di testi venga usato `nvi` invece di `vi`, e affinché nell'elenco di programmi per visualizzare i *files* `.htm` compaia `w3m` come prima opzione, subito prima di `links`, `lynx`.

Per uscire dal programma si usa il comando `:q`.

Text editor: `nvi`

Si tratta di un *editor* di testi di tipo modale, che usa cioè modalità diverse per inserire testo e compiere operazioni sul testo stesso. Premendo ESC si passa al *command mode*; da qui potranno essere usati i comandi descritti di seguito.

1. Inserire testo

Per entrare nella modalità *insert* bisogna premere

- `i` per inserirlo prima della posizione del cursore, oppure `a` per inserirlo dopo il cursore
- `I` per inserirlo all'inizio della riga con il cursore, oppure `A` per inserirlo alla fine di tale riga

2. Salvare e uscire

- `:w` per salvare il *file* (`:w` *nome_file* per salvarlo con un altro nome)
- `:wq` oppure `:x` per salvare e uscire
- `:q` per uscire e `:q!` per uscire senza salvare

3. Spostarsi nel testo

- Io trovo più comodi i tasti cursore, ma è possibile usare anche i tasti `h`, `j`, `k` e `l` per andare, nell'ordine, a sinistra, a destra, giù, su
- `0` (il numero "zero") per andare all'inizio della riga e `$` per andare alla fine
- `b` per tornare all'inizio della parola precedente e `w` per andare all'inizio della parola seguente; se si preme ad esempio `5w` si

andrà avanti di 5 parole

- `G` per andare all'ultima riga; se prima di `G` si indica un numero, si andrà a quella data riga (ad esempio `14G` porta alla quattordicesima riga)
- `^f` avanza di una schermata, `^b` indietreggia di una schermata
- `^d` avanza di mezza schermata, `^u` indietreggia di mezza schermata
- `^g` mostra in basso il nome del *file*, se è stato modificato dall'ultimo salvataggio, su quale riga ci si trova e su un totale di quante righe e a che punto del *file* ci si trova (in percentuale)

4. Cancellare testo

- `x` cancella il carattere su cui si trova il cursore (`4x` rimuove 4 caratteri a partire da quello su cui si trova il cursore), mentre `X` rimuove il carattere precedente il cursore
- `dd` cancella la riga su cui si trova il cursore; `3dd` cancella 3 righe a partire da quella col cursore
- `dw` rimuove la parola col cursore, a partire dal carattere in cui si trova il cursore stesso
- `D` cancella tutto partendo dal cursore fino alla fine della riga

5. Modificare testo

- `r` rimpiazza il carattere su cui si trova il cursore con il carattere che viene inserito subito dopo
- `cw` cambia la parola su cui si trova il cursore con quanto viene scritto dopo, partendo dal punto in cui si trova il cursore; per terminare la modifica e accettarla, si preme ESC. `cc` fa lo stesso ma con tutta la riga col cursore
- `cb` cambia la parola su cui si trova il cursore con quanto viene scritto dopo, partendo dal punto in cui si trova il

cursore e andando indietro fino all'inizio della parola; per terminare la modifica e accettarla, si preme ESC
- `u` annulla l'ultima modifica; se si preme ancora `u`, si annulla l'*undo*
- il "punto" `.` ripete l'ultima modifica fatta

6. Taglia/copia/incolla

- per copiare in memoria si usa `y` ("*yank*") in combinazione con i comandi di movimento per indicare cosa copiare (una parola, fino alla fine della riga, tre righe eccetera)
- il testo che viene cancellato (ad esempio una riga rimossa con `dd` come abbiamo visto prima) è in effetti tagliato e messo in memoria per poi essere eventualmente incollato
- per inserire il testo copiato in memoria si va prima di tutto sul punto desiderato e poi si preme `P` per incollare il testo in memoria prima del cursore oppure `p` per inserirlo dopo il cursore (il testo incollato resta in memoria)
- si può per esempio andare sulla prima riga da copiare, premere `mk` per marcare la riga, muoversi sull'ultima riga e digitare `y'k` per copiare in memoria o `d'k` per cancellare quanto selezionato prima della marcatura, andare sul punto dove bisogna incollare e premere `P` o `p` per incollare come visto prima
- è possibile premere `mm` per segnare l'inizio della riga su cui ci si trova, muoversi su un'altra riga e poi premere `d'm` per cancellare dal punto di prima fino alla riga in cui ci si trova, oppure `y'm` per copiare tale pezzo di testo in memoria (e poi incollarlo col solito comando `P` oppure `p`. È anche possibile usare `mk` al posto di `mm` e `d'm` e `y'm`
- `J` unisce la riga corrente con quella successiva

7. Ricerca e sostituzione

- Per cercare una parola dal punto in cui si trova il cursore in avanti, si digita `/`, poi di seguito la parola da cercare e poi si preme INVIO; per cercare invece all'indietro si usa `?`
- `n` ripete la ricerca nella stessa direzione in cui la si era fatta prima, mentre `N` la ripete nella direzione opposta
- Si possono usare espressioni regolari: `/cas.` cerca le occorrenze dei tre caratteri `cas` seguiti da un carattere qualunque
- Se serve cercare un `.` (il punto), farlo precedere da `\`. Per esempio, se devo cercare l'inizio di un capitolo nei miei sorgenti TROFF, devo usare `/\.CA`
- Un comando come `:%s/sbagliato/corretto/` si usa per sostituire in tutto il *file* (il carattere `%`) la parola `sbagliato` con la parola `corretto`; se alla fine, dopo la seconda barra, si aggiunge una `g` allora la ricerca con sostituzione verrà fatta su tutto quanto il *file*, e se si aggiunge invece `gc` allora per ogni occorrenza della parola da correggere verrà chiesto se confermare la sostituzione (premendo `y`) oppure no (premendo INVIO per passare all'occorrenza successiva oppure `q` per interrompere la ricerca)

8. *Files* multipli

È possibile anche aprire un secondo *file* mentre se ne sta editando uno. I metodi sono due:

- usando `:e` *nome_file* si apre il secondo *file*; premendo `^6` (oppure col meno comodo comando `:e#`) si passa dall'uno all'altro
- usando invece `:E` *nome_file*, il secondo *file* viene aperto in una mezza finestra in alto, per passare alla quale (e viceversa) occorre premere `^w`. Si può impostare l'altezza di

una delle due finestre posizionandoci su di essa e dando il comando `:res 20`, per esempio, per impostarla a 20 righe. Con questa modalità possiamo uscire da uno dei due *files* con il solito comando `:q`.

La cosa molto utile di questi due approcci è che permettono di copiare/tagliare delle righe in uno dei *files* per poi incollarlo sull'altro.

Accedere a reti IRC: `irssi`

È sufficiente lanciare il *client* IRC e poi dare il comando `/connect irc.freenode.net` per connettersi al *server* (in questo caso di Freenode). Una volta connessi, si può entrare in un canale con `/join #nome_canale`.

Adesso potete usare i seguenti comandi:

- `/set nick nuovo_nick` per impostare il *nick* da usare
- `/nick nuovo_nome` per cambiare il *nick*
- `/topic` per mostrare l'argomento del canale
- `/names` per elencare tutti gli utenti del canale
- `/whois nome_utente` per vedere informazioni su un dato utente
- `/msg nome_utente messaggio` per mandare un messaggio a uno specifico utente
- `/away messaggio` per impostarsi come "lontano dalla tastiera" mostrando il messaggio indicato
- `/part #nome_canale` per lasciare solo il canale in cui ci si trova senza disconnettersi dal *server*
- `/disconnect irc.freenode.net` per disconnettersi dal *server*
- `/quit` per disconnettersi e uscire
- `/quit` *messaggio* per disconnettersi e uscire mostrando il messaggio dato
- `/me azione`, invece, mostra l'azione indicata. Per esempio, se il mio nome utente è `pippo` e scrivo `/me rotola dal ridere`, il messaggio visualizzato sarà "pippo rotola dal ridere".

Ci sono poi alcuni comandi per i cosiddetti “operatori” ovvero quelli che, in un canale, hanno alcuni privilegi in più in quanto moderatori:

- `/ban` per vedere o impostare i *ban* di un canale
- `/kick` per mandar via un utente dal canale
- `/kickban` per mandarlo via e bannarlo
- `/unban` per togliere i *ban* a tutti

Per scaricare con il protocollo XDCC, se il programma lo supporta, si usa `/msg` *nome_bot* `xdcc send` *nome_pacchetto*. Quando si può ricevere, usare il comando `/dcc get` *nome_bot* per iniziare il *download*. Se si vuole cancellare il *download* in corso, usare `/msg` *nome_bot* `xdcc cancel`. Se invece il *download* non è ancora iniziato, possiamo rimuovere il pacchetto dalla coda con `/msg` *nome_bot* `xdcc remove` aggiungendo eventualmente subito dopo il numero del pacchetto per rimuovere solo quello.

Per cercare un *server* IRC si può andare su `https://netsplit.de`; qui si può anche verificare se quel dato *server* accetta connessioni SSL e su quale porta.

Per connettermi a queste reti, io ho scelto di usare `irssi`.

```
$ doas pkg_add irssi
```

Dopo averlo lanciato, per connettersi a un *server* si usa il comando `/connect` *nome_del_server*. Si può anche usare una connessione protetta: `/connect -tls` *nome_server* `6697`.

`irssi` utilizza diverse finestre per gestire le varie connessioni, i canali, i messaggi privati. È possibile passare da una finestra all’altra premendo ALT-*numero_finestra*; se non funziona, provate a premere ESC e poi di seguito il numero della finestra. In alternativa, c’è il comando `/win numero_finestra`. Con `/win list` potete visualizzare un elenco delle finestre. Con `^n` e `^p` si passa rispettivamente

alla finestra successiva e alla precedente. Invece con ESC-n si scorre di mezza schermata in avanti e con ESC-p di mezza schermata indietro.

Un comando molto comodo è `/hilight parola`, ove `parola` è il termine che volete venga evidenziato. Ciò è utile per tenere traccia di messaggi che riguardano quel dato tema, soprattutto quando vi sono molti utenti che scrivono tutti insieme. Per terminare questo comportamento basta indicare `/dehilight parola`.

Un altro comando che trovo molto utile è `/window hidelevel +joins +parts +quits`, che nasconde tutti i messaggi di utenti che si uniscono al canale o lo lasciano o escono dal programma IRC. Li si può anche nascondere di *default* col comando `/set window_default_hidelevel hidden joins parts quits`.

Si può aggiungere poi un altro *server* a quelli già presenti nell'installazione di base:

```
/network add libera
/server add -tls -network libera irc.libera.chat 6697
/network add -autosendcmd "/quote nickserv identify password; wait 2000" libera
```

Ora, dopo aver salvato con `/save`, si può usare il comando `/connect libera` per connettersi con SSL/TLS alla rete LiberaChat e identificarsi con NickServ.

Client FTP: `ftp`

OpenBSD ha già nell'installazione di base un *client* FTP, che si chiama proprio `ftp` e funziona in una finestra di terminale. È sufficiente digitare `ftp` *nome_server* (o in alternativa solo `ftp` e poi dare il comando `open` *nome_server*) per connettersi al *server* scelto. Eventualmente, dare poi il nome dell'utente e la password per poter accedere.

Non appena si accede a un sito, il programma indica se il trasferimento è impostato in modalità `ascii` oppure `binary`. Digitando una di queste due pariole chiave si imposta comunque la modalità corrispondente. La prima viene usata per i *file* di testo, la seconda per gli altri tipi di *file*, dalle immagini agli eseguibili agli archivi; se non si è sicuri, usare `binary`.

1. Navigare tra le directory

Questo programma funziona in pratica come una *shell*, quindi si usa:

- `ls` per mostrare il contenuto della *directory* remota
- `!ls` oppure `!ls -l` per mostrare quello della *directory* locale (il `!` indica che si vuole usare un comando in locale)
- `cd` *directory* per cambiare la *directory* remota
- `lcd` *directory* per mostrare o cambiare la *directory* locale

2. Pubblicare *files*

- `put` *nome_file* pubblica nella *directory* remota corrente il *file* *nome_file* che si trova nella *directory* locale corrente

- `put` *nome_file ./directory/altro_nome_file* pubblica nella *directory* remota `directory` il *file nome_file* che si trova nella *directory* locale corrente e lo rinomina in `altro_nome_file`
- `mput` *nome_file1 nome_file2* pubblica nella *directory* remota corrente i due *files* indicati, chiedendo conferma prima di ognuno dei due (è possibile usare anche delle *wildcards* e specificare ad esempio `*.txt`)

3. Scaricare *files*

- `get` *nome_file* scarica dalla *directory* remota corrente il *file nome_file* e lo mette nella *directory* locale corrente
- `get` *nome_file ./directory/altro_nome_file* scarica dalla *directory* remota corrente il *file nome_file* per metterlo nella *directory* locale indicata e lo rinomina in `altro_nome_file`
- `mget` *nome_file1 nome_file2* scarica dalla *directory* remota corrente i due *files* indicati, chiedendo conferma prima di ognuno dei due (è possibile usare anche delle *wildcards* e specificare ad esempio `*.txt`)

4. Gestire *files*

- `rename` *nome_file1 nome_file2* rinomina nella *directory* remota corrente il *file* `nome_file1` chiamandolo `nome_file2`
- `delete` *nome_file* cancella il *file* indicato dalla *directory* remota corrente
- `mkdir` *nome_directory* crea nella *directory* remota corrente una nuova *directory* con il nome specificato
- `rmdir` *nome_directory* rimuove dalla *directory* remota corrente la *directory* indicata

5. Uscire dal programma

- `close` chiude la connessione col *server* FTP restando però nel programma
- `exit` o `quit` chiude la connessione ed esce dal programma

Browser da terminale: `w3m`

1. Visitare un sito

- si lancia il programma con `w3m` seguito dall'indirizzo della pagina da visitare
- per aprire invece una nuova URL, si preme `SHIFT-u` e poi si digita l'indirizzo

2. Navigazione nella pagina

- per spostarsi in basso nella pagina basta premere la barra spaziatrice oppure `+`
- per spostarsi in alto si preme semplicemente `b` oppure `-`

3. Apertura dei *link*

- premendo TAB si seleziona il *link* successivo
- premendo `SHIFT-TAB` oppure `^u` si torna al *link* precedente.

Una volta evidenziato un *link*, con INVIO lo si seleziona. Premendo `SHIFT-b` si torna alla pagina precedente.

Con `SHIFT-l` vengono mostrati i soli *link* presenti nella pagina (`SHIFT-b` per tornare indietro). Si può anche premere `ESC-m` per farli apparire in un menu (`^c` per chiuderlo).

4. Ricerca in una pagina

Per cercare una parola nella pagina si preme `/` (ricerca in avanti) o `?` (ricerca all'indietro).

Esiste anche una ricerca incrementale che si attiva con ^s (in avanti) o ^r (all'indietro). Per cancellare la ricerca che si stava impostando si preme ^c.

5. Altri comandi

Per aprire la pagina di aiuto con i comandi da usare, SHIFT-h (SHIFT-b per tornare indietro).

Per aprire un menu di opzioni premere o (SHIFT-b per chiuderlo).

Per uscire dal programma si preme SHIFT-q (solo q per uscire chiedendo conferma dell'uscita).

Da notare che w3m è in effetti un *pager*, e per questo offre alcune funzioni interessanti:

- premendo v si visualizza il sorgente HTML della pagina *web* che si sta vedendo (e ripremendo il tasto si torna alla visualizzazione normale)
- con E si apre l'*editor* definito nel *file* di configurazione per modificare la pagina che si sta visualizzando (solo se questa si trova in locale)

6. Navigazione con schede

w3m offre anche la possibilità di navigare con delle *tab*. SHIFT-t duplica la pagina corrente in una nuova scheda. Per aprire invece il *link* attualmente selezionato in una nuova scheda si preme ^t.

ESC-t mostra un menu di navigazione per le varie schede (^c per chiuderlo). Si può comunque andare alla scheda precedente o alla successiva rispettivamente con { (SHIFT-[) e } (SHIFT-]).

Per chiudere la scheda corrente si usa ^q.

Posta elettronica: `aerc`

```
$ doas pkg_add aerc abook
```

Io uso questo programma con la mia casella di posta su GMail, col protocollo IMAP. Innanzitutto ho quindi dovuto accedere a Gmail da *web*, impostare la mia casella di posta con questo protocollo invece di POP3, poi andare nella sezione “Sicurezza”, attivare la protezione in due passaggi e infine impostare una *password* apposita per le *app* “meno sicure”.

Una volta lanciato il programma per la prima volta, rispondendo ad alcune domande si imposta già il *file* per accedere alla posta, che però va modificato un pochino. Apro quindi il *file* `~/.config/aerc/accounts.conf` e controllo che sia scritto:

```
source = imaps://mia_email%40gmail.com:password@imap.gmail.com:993
outgoing = smtps://mia_email%40gmail.com:password@smtp.gmail.com:465
default = INBOX
from = "Antonio Frecentese" <mia_email@gmail.com>
cache-headers = true
```

Ora posso semplicemente lanciare `aerc` e godermi la posta su GMail con IMAP.

È possibile modificare alcune impostazioni del programma per renderlo più piacevole alla vista e più comodo da usare. Ecco allora che nel *file* `~/.config/aerc/aerc.conf`

• ho modificato la riga per il formato della data cambiandola in

```
timestamp-format=2006-01-02 15:04
```

- ho modificato la larghezza della barra laterale per ampliarla:
  `sidebar-width=25`
- ho inserito l'opzione `sort=from -r date`
- ho impostato il raggruppamento delle conversazioni:
  `threading-enabled=true`
- ho impostato `editor=nvi` per poter usare anche giapponese, cinese, arabo...
- ho aggiunto altri campi per le email da comporre:
  `header-layout=To,Cc|Bcc,From,Subject`
- ho impostato il comando per la rubrica:
  `address-book-cmd=abook --mutt-query '%s'`
- ho aggiunto nella sezione `[filters]` la riga `text/html=w3m -T text/html -o display_link_number=1` per poter vedere le email in formato HTML usando `w3m`

Ho creato la *directory* `~/.config/aerc/stylesets`, vi ho copiato dentro lo stile `default` con un altro nome e ho specificato di andare a usare quello come stile; in questo *file* ho poi semplicemente indicato di impostare col colore blu i messaggi non ancora letti, in modo da trovarli più agevolmente.

Per finire, nel *file* `~/.config/aerc/binds.conf` ho commentato la riga con `<tab> = :next-field<Enter>` per fare in modo che il tasto TAB, anziché farmi passare al campo successivo, cercasse con `abook` l'indirizzo di destinazione e lo completasse, e nella sezione `[view]` ho aggiunto la seguente riga per far sì che, premendo `a` leggendo un'email, il mittente venisse aggiunto alla rubrica di `abook`:

```
a = :pipe -m abook --add-email-quiet<Enter>
```

I primi semplici comandi da usare sono:

- per muoversi da un messaggio all'altro nella lista: `j` o `k`, oppure i tasti della freccia verso l'alto o verso il basso

- per muoversi da una cartella all'altra nella lista laterale: `SHIFT-j` o `SHIFT-k`, oppure i tasti della freccia verso l'alto o verso il basso tenendo premuto CTRL
- INVIO per aprire una email
- `q` per tornare poi alla lista dei messaggi
- `s` per aprire/chiudere un riquadro di anteprima del messaggio (`S` per farlo apparire a destra invece che sotto); attenzione, spesso sfarfalla piuttosto fastidiosamente
- `C` per comporre un messaggio
- `d` per cancellare il messaggio (`D` per farlo senza dover confermare)
- `r` per rispondere al messaggio
- `:mv` *nome_cartella* per spostare il messaggio in quella data cartella
- `:q` per uscire

Una volta che si preme INVIO per aprire un messaggio, se questo è disponibile sia in formato testo che HTML si può premere `^-freccia su` oppure `^-freccia giù` per scegliere uno dei formati disponibili. Lo stesso per scegliere uno degli eventuali allegati.

Quando si compone una email, al termine si preme ESC e poi `:wq` e INVIO per uscire dall'*editor* (in fin dei conti stiamo usando `nvi`, no?), e a questo punto compaiono alcune opzioni per allegare *files* (uno alla volta; si può anche rimuoverli dalla lista, sempre uno alla volta), inviare, ritardare l'invio o non inviare.

Rubrica contatti: `abook`

```
$ doas pkg_add abook
```

Dopo averlo lanciato con `abook`, è sufficiente premere `a` per inserire un nuovo contatto: in basso compare subito un *prompt* che richiede il nome completo del contatto da salvare, e quando si preme INVIO viene richiesto l'indirizzo email da aggiungere alla rubrica; una volta scritto l'indirizzo e premuto INVIO, siamo a cavallo.

I vari campi sono numerati, e per modificarli o aggiungerli è sufficiente premere il loro numero. Premendo poi i tasti freccia verso destra o verso sinistra si può passare alle altre schede per aggiungere ulteriori dettagli (indirizzo, numero di telefono...).

Prima di uscire dal programma, ricordarsi di premere `w` per salvare i dati.

Player audio: `cmus`

Per ascoltare musica, data la scarsa potenza del PC, scelgo di installare un *player* da terminale:

```
$ doas pkg_add cmus
```

Dopo averlo lanciato con `cmus`, è sufficiente premere `5` per andare nel suo *browser* interno, scorrere le varie cartelle e premere `a` per selezionare i *files* e poi `:save` per salvare la libreria. Poi premendo `2` si va alla libreria e, una volta scelto il *file* si preme INVIO per suonarlo (`c` per mettere e togliere la pausa, le frecce a dx/sx per spostarsi di 10 secondi, − per abbassare il volume e + per alzarlo). Facile, no? Per eliminare invece dalla *playlist* un *file*, selezionatelo e premete `D`.

Questo programma è anche in grado di aprire *stream* audio: per esempio provate a premere `:` e poi inserire `a http://anonradio.net:8000/anonradio` (`:` fa entrare nel *command mode*, mentre la `a` sta per *add*).

Terminal multiplexer: `tmux`

Si tratta di un programma molto utile (e già presente nell'installazione di base) per gestire il lavoro di chi usa tante finestre di terminale contemporaneamente. Lo si lancia dando il comando `tmux`. In basso compare una comoda barra di stato che indica nell'ordine, partendo da sinistra:

- `[0]`: il nome della sessione. Di *default*, infatti, sono numerate partendo dallo zero
- `[0:ksh*]`: lo `0` indica che si è nella prima finestra di questa sessione, `ksh` indica che il processo attualmente in funzione è appunto la *shell* `ksh` mentre l'asterisco indica che questa è la finestra che stiamo guardando in questo momento
- il nome del dominio
- l'ora e la data

Da questa situazione, se diamo il comando `exit` usciamo dal programma.

Premendo `^b` e poi `?` verrà mostrata una schermata con i vari comandi disponibili; potete premere ESC per chiuderla e tornare al programma.

1. I pannelli

Una delle cose che faccio solitamente con questo PC è scrivere un documento per l'impaginatore `troff`. Uso quindi due finestre di terminale, una per l'*editor* `nvi` e una per dare i comandi di compilazione. Voi potreste fare lo stesso per scrivere i sorgenti dei vostri programmi e per lanciare compilatore e *linker* per produrre i vostri capolavori, per

esempio. Ebbene, invece di aprire due finestre di terminale e ridimensionarle posso aprirne una sola, ingrandirla per farle occupare tutto lo schermo (dato che di per sé è piccolo, sfruttiamolo tutto!) e poi lanciare `tmux` per usarle entrambe. Come sarebbe, "entrambe"? Semplice: dividendo la finestra di `tmux` in due pannelli.

Per prima cosa, si preme `^b`, che è la sequenza di tasti usata per richiamare l'attenzione di `tmux`, dopodiché basta premere `"` se si vuole dividere la finestra in due pannelli in senso verticale, uno sopra l'altro, oppure `%` per dividerla in due in senso orizzontale, un pannello a destra dell'altro. Ora è come se avessimo due finestre di terminale aperte, visibili contemporaneamente.

Ecco alcuni comandi da usare coi pannelli:

- `^b` seguito da un tasto freccia: ci si sposta nel pannello che si trova nella direzione della freccia
- `^b` seguito da `q`: vengono mostrati per qualche istante dei numeri che identificano i vari pannelli
- `^b` seguito da `o`: si passa al pannello seguente
- `^b` seguito da `{` o da `}`: scambia il pannello corrente rispettivamente con quello successivo o con quello precedente
- `^b` seguito da CTRL-freccia: ridimensiona il pannello attuale andando nella direzione indicata dal tasto freccia (comodo per esempio per ridurre uno dei pannelli all'altezza di poche righe, quel tanto che basta per poter leggere l'*output* di un compilatore e lasciare leggibile il grosso del sorgente su cui si sta lavorando)
- `^b` seguito da `x`: chiude il pannello corrente (viene chiesta conferma: basta premere `y` per confermare o `n` per non chiudere il pannello)

Per uscire da un pannello si può anche dare il comando `exit`.

2. Le finestre

Un'altra funzione utile di `tmux` è la possibilità di usare diverse finestre. È un po' come usare dei *desktop* virtuali, ma tutto dentro un'unica finestra di terminale. Quindi mentre i pannelli possono essere ampliati o ristretti e sono comunque sempre visibili contemporaneamente, le finestre non possono essere modificate di dimensioni e sono visibili solo una per volta.

- `^b` seguito da `c`: apre una nuova finestra
- `^b` seguito da `,`: rinomina la finestra (in modo che il suo nome compaia nella barra di stato sottostante e sia più facilmente comprensibile cosa vi si stia facendo)
- `^b` seguito da `n`: va alla finestra successiva
- `^b` seguito da `p`: va alla finestra seguente
- `^b` seguito da un numero tra 0 e 9: va alla finestra corrispondente a quel numero
- `^b` seguito da `w`: apre una lista delle finestre attive con anche gli eventuali pannelli aperti in ognuna di esse, permettendo di vedere cosa ci sia in ogni finestra e pannello e, premendo INVIO, di andare in quella finestra e pannello (se si preme ESC si resta invece nella finestra attuale). Se invece di INVIO si preme `x`, viene chiesto se eliminare la finestra in quel momento evidenziata (basta premere `y` per confermare l'operazione)

3. Le sessioni

Una comoda funzione di `tmux` è legata all'uso delle sessioni. Immaginate di lavorare in una finestra di terminale su un vostro programma, avere scritto molte linee di codice e

lanciato il compilatore, ma le operazioni vanno per le lunghe. Supponete ora di dovervi dedicare ad altro, lasciando però proseguire il lavoro del compilatore. Invece di aprire un'altra finestra di terminale, se siete già all'interno di `tmux` potete ricorrere alle sessioni.

La prima cosa a fare è dare un nome alla sessione corrente, che descriva ciò che si sta facendo (in caso contrario, la sessione avrà un numero partendo dallo zero): `^b` seguito da `$` mostra nella barra di stato l'attuale numero o nome di sessione, da modificare con quello nuovo, sempre di una sola parola. Notare che si può anche far partire direttamente il programma facendogli creare una sessione di un dato nome: `tmux new -s nomesessione`.

Ora, se si preme `^b` e poi `d`, si verrà slegati dalla sessione corrente e si tornerà alla *shell*; `tmux` lo segnalerà col messaggio "*detached*" e indicando il nome della sessione in cui ci si trovava).

Per ritornare nella sessione basta dare il comando `tmux attach-session -t nomesessione`.

Si può mostrare una lista delle sessioni con `^b` seguito da `s`: si avrà una lista delle sessioni in cui navigare con i tasti freccia su e giù, vedendo nella parte inferiore dello schermo l'attuale *output* della sessione evidenziata, e premendo INVIO si andrà a quella sessione. Premendo ESC, invece, si chiuderà la lista e si tornerà alla sessione in cui ci si trovava.

Anche dalla finestra di terminale si può vedere un elenco delle sessioni, col loro nome o numero identificativo: basta dare il comando `tmux ls`.

EXTRA: l´*editor* `ed`

Si tratta di un *editor* molto particolare, presente nell'installazione di base. Lo si lancia con `ed`, ma io preferisco qualcosa tipo `ed -p 'ed > '` affinché venga mostrato un *prompt* prima di inserire un comando; in questo modo è più semplice per me capire se `ed` sta attendendo un testo o un comando (se non lo si fa, infatti, `ed` non mostra alcunché).

Per inserire del testo, si entra innanzitutto in modalità inserimento premendo `i` per inserire del testo oppure `a` per appenderlo alla fine del *buffer*, e premendo poi INVIO. Si passa poi a scrivere il proprio testo. Per finire si inserisce un `.` (punto) da solo su una sua riga, premendo poi INVIO. Fatto questo, si può dare il comando `w` per salvare il *file*, o eventualmente `w` *nome_file* come equivalente di "salva con nome". Da notare che `ed` salverà il *file* e mostrerà il numero di caratteri da lui scritti. Per uscire, basta dare il comando `q` e premere INVIO.

Per caricare un *file* già salvato, una volta lanciato `ed` si dà il comando `e` *nome_file*. È possibile anche usare `r` *nome_file*: in questo caso, anziché svuotare il *buffer* e caricare il nuovo *file* in memoria, `ed` lo aggiungerà in coda al testo eventualmente già presente in memoria. Col comando `f` è possibile vedere il nome del *file* che si sta modificando.

`ed` non mostra molto, in genere, e anche solo per visualizzare il contenuto del *file* caricato in memoria occorre dare un comando. La sintassi è *pagina_iniziale*`,`*pagina_finale*`p`. Per esempio, `1,5p` mostrerà

le prime 5 righe del *file* caricato in memoria. Una sintassi alternativa usa la lettera `n` invece della `p`: con tale comando, all'inizio delle righe verrà anche riportato il loro numero all'interno del *file*.

Altri caratteri particolari sono `$`, che indica l'ultima riga del *buffer*, e il carattere `.` (punto), che indica la riga corrente. Per "riga corrente" si intende l'ultima che ha subìto un dato comando, che sia quello di stampa o di modifica. È bene fare molta attenzione a questo, perché sapere in quale riga ci si trova, con `ed`, è fondamentale. Alcuni comandi modificano il valore di `.` mentre altri non lo fanno. In ogni momento si può chiedere in quale riga ci si trovi col comando `.=`.

Naturalmente, si può anche scrivere `5p` per mostrare solo la riga numero 5, o `.-1p` per mostrare la riga precedente, e così via.

Per cancellare una riga si usa il comando `d`. La sintassi è la stessa del comando `p`, quindi per esempio `4,$d` cancella le righe dalla numero 4 in poi fino alla fine del *file*. Il valore di `.` diventa la riga dopo l'ultima cancellata, a meno che questa non sia l'ultima del *file* poiché in questo caso diventa lo stesso di `$`.

Si può anche caricare un testo all'inizio del *file* con un comando tipo `0r` *nome_file*, o aggiungere testo semplice all'inizio del *buffer* con `0a`.

Per sostituire del testo si ricorre al comando `s`. Ad esempio, `5s/proba/prova/` indica "alla riga numero 5 sostituisci 'proba' con 'prova'". Anche qui è possibile usare un intervallo di righe come con `p`, ma c'è una cosa cui fare attenzione: se non avviene nessuna sostituzione, il valore di `.` non cambia. Si può anche scrivere `s/...//`, ossia sostituire quella data parola o stringa di caratteri con "nulla", quindi cancellarla: `s/xx//p` rimuove dalla riga corrente i due

caratteri “xx” e poi mostra la riga modificata.

Il comando di sostituzione cambia solo la prima occorrenza della sequenza di caratteri indicata. Per indicarle tutte, occorre specificare `s/.../.../gp`: la `g` finale indica “globale”, in tutto il *file*.

La semplice ricerca, invece, senza sostituzione, si ottiene con solo le due barre: `/prova` cerca la parola “prova”. Da notare che, una volta trovata, il valore di `.` viene impostato su quella data riga e la riga stessa viene mostrata. La ricerca parte da `.+1`, va fino alla fine del *file* e poi riparte dalla riga 1 fino alla riga `.`; se non trova nulla, compare un messaggio di errore, ovvero un semplice `?`. Per ripetere l’ultima ricerca fatta, si usa `//`.

La ricerca può essere fatta anche all’indietro, usando il carattere `?` invece di `/`: `?prova?`. La ricerca può essere ripetuta all’indietro con `??`.

Infine, il comando `c` è usato per cambiare un dato intervallo di righe con delle altre righe. Per esempio, `.,$c` si usa per cambiare il testo dalla riga seguente fino alla fine del *file* con quanto verrà digitato dopo il comando (fino al `.` come fosse del testo nuovo). Ovviamente, se prima di `c` viene specificata una sola riga, solamente quella verrà rimpiazzata.

È anche possibile inserire del testo: per esempio `/`*stringa*`/i` permette di inserire *prima della riga* che contiene la stringa *stringa* del testo a piacere, fino al solito `.` che termina l’inserimento del testo. Se non è specificato un numero di riga, viene ovviamente sottintesa quella corrente. Il valore di `.` diventa quello dell’ultima riga inserita.

Un ulteriore comando utile è `m`, che serve per spostare del testo (taglia/incolla). La sintassi riporta tre numeri di riga: *riga_iniziale*`,`*riga_finale*`m`*dopo_questa_riga*. Per esempio, `1,3m$` sposterà le prime 3 righe alla finel del *buffer*. La riga

corrente diventa l'ultima del blocco che è stato spostato.

`g` esegue un certo comando su tutte le righe del *buffer* che contengono una data stringa. Ad esempio, `g/prova/p` mostra tutte le righe che contengono la parola "prova". Oppure, `g/prova/s//esercizio/p` trova in tutto il *buffer* le righe che contengono "prova" e al loro interno sostituisce tale parola con la parola "esercizio", per poi mostrare tali righe. Una versione "interattiva" di questo comando usa la `G` invece che la `g`.

`v` funziona come `g` ma al contrario, nel senso che esegue il comando indicato sulle righe che NON contengono la stringa che segue `v`. Anche qui si ha la variante "interattiva" con `V` al posto di `v`.

Si può usare il carattere `!` per inserire il risultato di un comando della *shell*: per esempio `r !ls` inserirà nel *buffer* il risultato del comando `ls` nella cartella corrente.

Il comando `u` serve come *undo*, quindi riporta alla situazione precedente l'ultimo comando (anche il valore di `.` viene ripristinato alla riga precedente). Un ulteriore `u` riesegue di nuovo l'ultimo comando (non ci sono quindi infiniti *undo*).

È importante, infine, notare che ci sono dei caratteri molto particolari da usare:

- il carattere `.` all'interno di una ricerca indica "qualsiasi carattere", quindi `/x.y/` significa "una riga che contiene una 'x', un carattere qualsiasi e una 'y'"
- il carattere `^` indica l'inizio di una riga, quindi `/^prova/` riporta la riga solo se la parova "prova" si trova all'inizio della riga stessa
- il carattere `$` è l'esatto opposto di `^`: `/prova$/` trova la riga solo se la parola "prova" si trova alla fine della riga stessa

- `a*` significa “un numero qualunque di ’a’”
- il carattere `&` indica “tutto ciò che si trova nella parte di sinistra”, perciò `s/.*/(&)/` significa “prendi tutta la riga (ovvero quanto indicato da ’.*’, ogni carattere per tutte le volte che compare) e sostituiscila con sé stessa ma racchiusa tra parentesi”
- per sostituire questi caratteri particolari col carattere medesimo, bisogna farlo precedere da una barra inversa, quindi ad esempio `s/ampersand/\&/` sostituisce la parola “ampersand” col simbolo &

Altri programmi usati

In questa sezione presento programmi installati in precedenza ma poi sostituiti da altri. In questa maniera posso tenere una sorta di *backup* di informazioni, qualora volessi in futuro reinstallarli.

1. Accedere a reti IRC: `ircII`

```
$ doas pkg_add ircii
```

Imposto alcune opzioni nel *file* `.ircrc`:

```
^set display off
set IRC_ENCODING UTF-8
set INPUT_ENCODING UTF-8
set DISPLAY_ENCODING UTF-8
set colour on
set novice off
nick chakuari
set display on
window new
window name crap
window shrink 5
window refnum 1
ignore * crap
#
# miei messaggi
#
on ^send_public * if (C == [$0]) \
    { echo $Z ^B^C10<$(N)>^O $1- } \
    { echo $Z ^B^C10<$(0)>^O $1- }
on ^send_msg * echo $Z -> ^B^C10*$(0)*^O $1-
#
```

```
# messaggi altrui
#
on ^public * echo $Z ^B^C2<$(0)>^O $2-
on ^msg * echo $Z ^B^C2*$(0)*^O $1-
```

Ora i messaggi sono un po' più leggibili, dato che i *nick* sono a colori. Ricordiamo che per impostare ad esempio `^B` occorre inserire un vero e proprio carattere di controllo, per cui con `nvi` dovremo premere `^V` e poi `B`. `^V` seguito da `I` è invece il tasto TAB.

Dopo aver lanciato il programma, per connettersi a un *server* si usa il comando `/server` *nome_del_server*. Si può anche usare una connessione protetta: `/server -ssl` *nome_server*`:6697`.

Si può creare una finestra separata con

```
/window new
/window name nome_per_la_finestra
/window level all
```

In questa maniera i vari messaggi del *server* e/o di disturbo o quelli agli altri utenti andranno in quella finestra. Posso anche usare quella finestra per visualizzare ad esempio l'*help* in linea, e poi passare alla finestra "normale" per le conversazioni. Per passare da una finestra all'altra si preme prima `^x` e poi `n` oppure `p`.

Con ESC-n si scorre in avanti nella finestra di visualizzazione, mentre con ESC-p si torna indietro.

Indicando `/ignore * crap` è possibile fare in modo che non vengano visualizzate le notifiche di utenti che si aggiungono al canale o che lo lasciano, cosa che rende più agevole la lettura in certi canali.

2. Accedere a reti IRC: `catgirl`

```
$ doas pkg_add catgirl
```

Creo il *file* `./config/catgirl/libera`:

```
nick = mio_nick
real = Antonio Frecentese
host = irc.libera.chat
```

Ora mi basta lanciare `catgirl libera` per connettermi all'*host* indicato.

Questo programma ha il vantaggio di connettersi solo con connessioni protette ed è discretamente leggero e veloce, ha la colorazione dei *nick* e finestre diverse (si passa dall'una all'altra con `^n` e `^p`), ma non si può usare XDCC per scaricare *files*.

3. Posta elettronica: `neomutt`

```
$ doas pkg_add neomutt abook
```

Io uso questo programma con la mia casella di posta su GMail, col protocollo IMAP. Innanzitutto ho quindi dovuto accedere a Gmail da *web*, impostare la mia casella di posta con questo protocollo invece di POP3, poi andare nella sezione "Sicurezza", attivare la protezione in due passaggi e infine impostare una *password* apposita per le *app* "meno sicure".

Ho poi inserito nel *file* `~/.config/mutt/muttrc` i parametri per la connessione e alcune personalizzazioni:

```
# utente
set from = "mia_email@gmail.com"
set realname = "Antonio Frecentese"

# GMail
set imap_user = "mia_email@gmail.com"
set smtp_url="smtps://mia_email@smtp.gmail.com:465/"
set smtp_authenticators = "plain"
set imap_pass = "mia_password_app"
set smtp_pass = "mia_password_app"

# forzare TLS
```

```
set ssl_starttls = yes
 set ssl_force_tls = yes

 # caselle di posta
 set folder = "imaps://imap.gmail.com:993/"
 set spoolfile = "+INBOX"
 set postponed = "+[Gmail]/Bozze"
 set record = "+[Gmail]/Posta inviata"
 set trash = "+[Gmail]/Cestino"
 set imap_check_subscribed

 # dove mettere le cose
 set header_cache = "~/.config/mutt/headers"
 set message_cachedir = "~/.config/mutt/bodies"
 set certificate_file="~/.config/mutt/certificates"
 unset record

 # sidebar
 set sidebar_short_path
 set sidebar_folder_indent
 set sidebar_format = "%B %* %N / %S"
 set mail_check_stats
 bind index,pager \Cp sidebar-prev
 bind index,pager \Cn sidebar-next
 bind index,pager \Ca sidebar-open
 bind index,pager B sidebar-toggle-visible
 bind index "," imap-fetch-mail

 set move=no# non va a un nuovo messaggio dopo aver
letto
 #set edit_headers# mostra headers quando componi
 set askcc
 set askbcc
 set fcc_attach# salva allegati nel body
 set forward_format = "Fwd: %s"
 set mime_forward = "ask-no"# inoltra allegati come
```

```
parte del messaggio?
 set include # include messaggio nelle risposte
 set fast_reply
 set charset = "utf-8"
 set sort_re
 set sort = reverse-threads
 set sort_aux = last-date-received
 set editor = "nvi"
 set pager_context=3
 set pager_stop=yes# si ferma alla fine del
messaggio
 set uncollapse_jump

 # mostra solo headers interessanti e nascondi gli
altri
 ignore *
 unignore from date subject to cc bcc
 unignore organization organisation x-mailer: x-
newsreader: x-mailing-list:
 unignore posted-to:

 # personalizza indice
 set date_format="%y-%m-%d %H:%M"
 set index_format="%2C | %Z [%d] %-20.20F (%-4.4c)
%s"

 # mailcap
 set mailcap_path = "~/.config/mutt/mailcap"
 # apri allegati con <invio>
 bind attach <return> view-mailcap
 # salva allegati in una cartella
 macro attach s '<save-entry>
<bol>~/Downloads/<eol>' 'save attachment'
 auto_view text/html
 alternative_order text/plain text/html
```

```
# abook
set query_command="abook --mutt-query '%s'"
macro index,pager a "<pipe-message>abook --add-
email-quiet<return>" "Add this sender to abook"
bind editor <Tab> complete-query

# colori (fronte sfondo)
color index yellow default '.*'
color index_author red default '.*'
color index_number blue default
color index_subject cyan default '.*'
# per nuove email
color index brightyellow black
color index_author brightred black
color index_subject brightcyan black
# colori headers:
color header blue default
color header brightmagenta default
color header brightcyan default
color header brightwhite default
mono bold bold
mono underline underline
mono indicator reverse
mono error bold
color normal default default
color indicator brightblack white
color sidebar_highlight red default
color sidebar_divider brightblack black
color sidebar_flagged red black
color sidebar_new green black
color normal brightyellow default
color error red default
color tilde black default
color message cyan default
color markers red white
color attachment white default
```

```
color search brightmagenta default
color status brightyellow black
color hdrdefault brightgreen default
color quoted green default
color quoted1 blue default
color quoted2 cyan default
color quoted3 yellow default
color quoted4 red default
color quoted5 brightred default
color signature brightgreen default
color bold black default
color underline black default
color normal default default
color body brightred default
# indirizzi email
#color body brightblue default
```

Ora bisogna creare il *file* `~/.config/mutt/mailcap`:

```
text/html; w3m -I %{charset} -T text/html; copiousoutput;
text/plain; cat %s; copiousoutput;
application/pdf; zathura %s; needsterminal;
```

Questo mi consentirà di aprire gli allegati PDF con `zathura` e le email in formato HTML con `w3m`.

Non dimentichiamo poi di creare in `~/.config/mutt/` le cartelle `headers`, `bodies` e `certificates`.

Una volta lanciato il programma col comando `neomutt`, in alto compare un menu con i tasti principali, e premendo `?` compare la lista dei comandi attualmente associati e usabili. Come si può vedere nel *file* `muttrc`, a quelli standard del programma ho aggiunto solo `B` per mostrare o nascondere un riquadro a sinistra con le varie cartelle di posta, `^n` per andare alla cartella successiva nell'elenco, `^p` per andare a quella precedente e `^a` per spostarsi in tale cartella e aprirne il contenuto, `s` nella finestra degli allegati (vi si accede con `v`)

per salvare l'allegato stesso. Inoltre, ho modificato `a` per fare in modo che con quel tasto il mittente dell'email selezionata venga aggiunto a un *file* di rubrica mediante `abook`: scrivendo un'email e premendo TAB, potrò scegliere a chi inviarla dalla rubrica invece che digitare l'indirizzo a memoria.