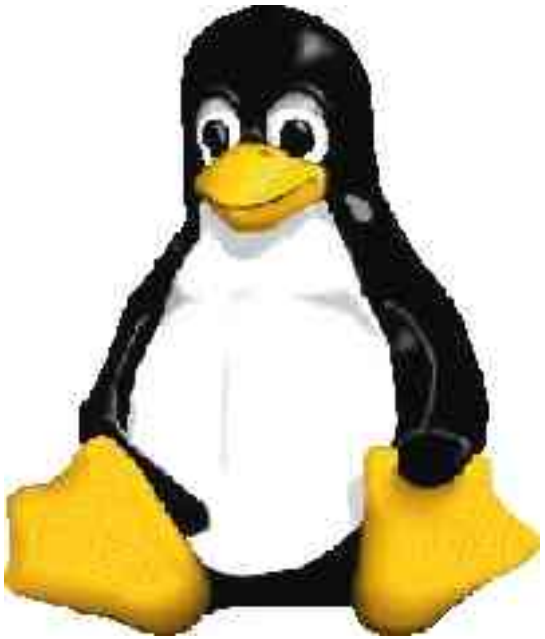
Linux Kernel (Hac|Ma)king
make it simple

Come compilarsi da soli il proprio kernel



Pallotron – 16 Febbraio 2004

Centro Culturale Zo' di Catania (piazzale ASIA)

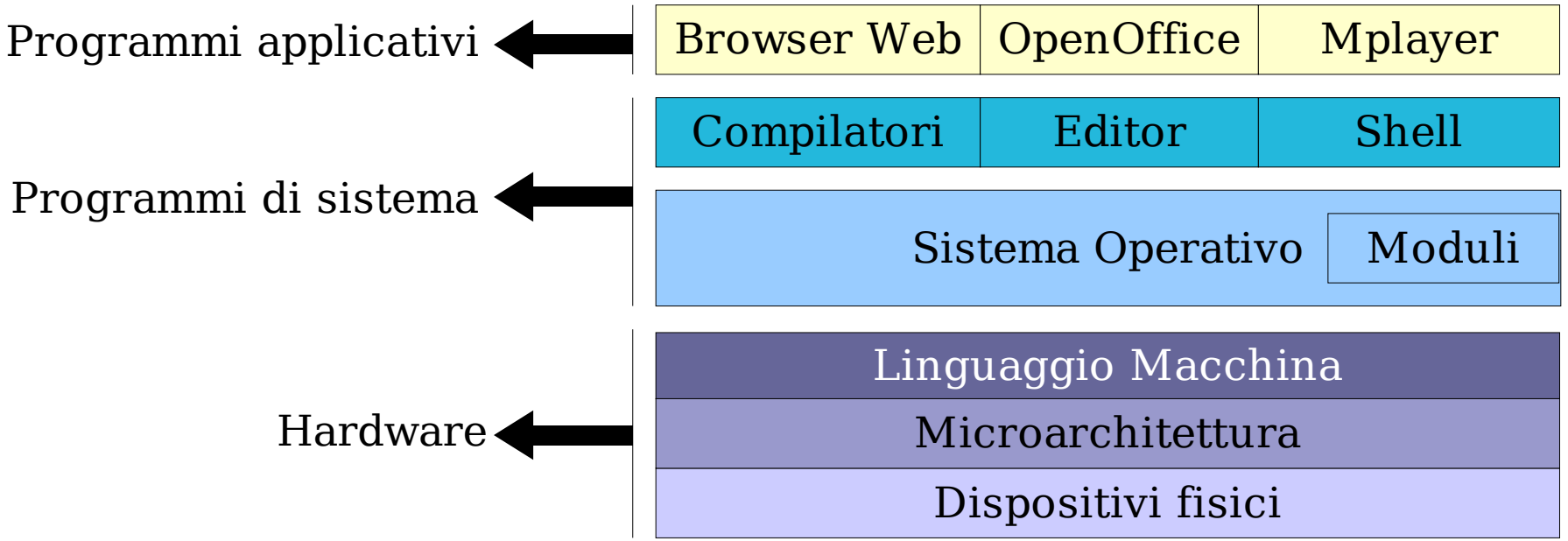
Sommario

- Cosa e' un kernel? Perche' e' bene compilarmene uno?
- Dove lo prendo?
- Requisiti necessari alla compilazione;
- Overview della compilazione;
- Ottenere ed estrarre il kernel;
- Applicare una patch;
- Configurazione;
- Compilazione;
- Installazione: il LI.LO.;
- I device driver sotto Linux;
- Cenni: Come compilare un kernel su un'altra macchina.

Cosa e' un kernel?

E' uno strato software che si colloca tra l'hardware di una macchina e i programmi applicativi e/o di sistema da essa eseguiti. Tutti i kernel implementano due funzionalita' che sono scorrelate tra di loro:

- Estendono la macchina sulla quale girano (es. Filesystem Unix);
- Gestiscono opportunamente le risorse della macchina (es. device di i/o), condividendole rispetto al tempo ed allo spazio;



Perche' e' bene compilarsene uno da se'? - 1/3

Ci sono innumerevoli motivi! :)

- I kernel di default delle distribuzioni sono generalmente carichi di funzionalita' da noi non richieste;

- I kernel di default delle distribuzioni sono vecchi;

- Potremmo avere la necessita' di compilarci un kernel minimale per una macchina che deve assolvere a compiti particolari (es. Firewall);

Perche' e' bene compilarsene uno da se? - 2/3

- Potremmo avere la necessita' di fare del cross-compiling;

- L'acquisto di un nuovo dispositivo hardware potrebbe portarci alla necessita' di ricompilare il nostro kernel.

- Le procedure di compilazione assistita delle distribuzioni o i kernel in forma binaria sono sicuri?

Perche' e' bene compilarsene uno da se? - 3/3

- E' sempre bello capire come funzionano le cose e fare tutto da se! A costo di rimanere inchiodati davanti al pc per ore e ore! :)

Dove lo prendo?

http://www.kernel.org (via http/ftp/rsync)

Normalmente il kernel viene fornito sotto forma di archivi tar compressi oppure sotto forma di patch cumulative da applicare a versioni precedenti presenti sul disco fisso.

Di solito trovate tutto quello che vi interessa nella directory **/pub/linux/kernel.**

E' buona regola depositare i tar e le patch dentro la directory **/usr/src/**

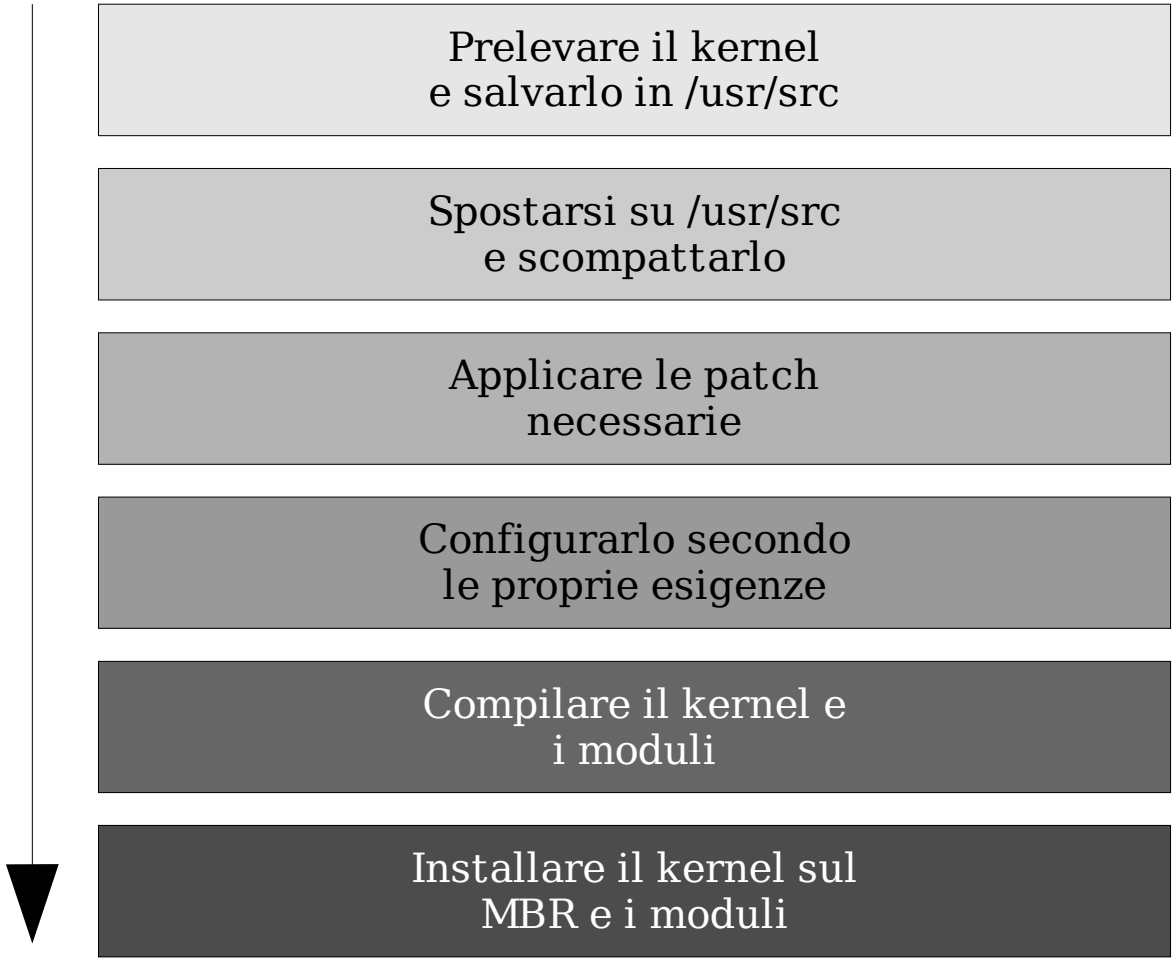
Requisiti e software necessari per la compilazione

- Avere installate le librerie **NCURSES**;
- Avere un ambiente di compilazione sano (leggi **make** + **gcc**);
- Per i kernel della serie 2.6 occorre avere installati i nuovi programmi per la gestione dei moduli (**module-init-tools**).
- Dovete avere i privilegi di amministrazione della macchina in questione! :)

Tutti i sistemi hanno pacchetti binari preconfezionati. Ad esempio, su debian, e' necessario dare questo comando e avrete tutto il necessario per compilare un kernel:

```
# apt-get install module-init-tools libncurses5-dev make gcc
```

Overview: la compilazione del kernel in step



Estrazione del kernel - 1/2

Supponendo che abbiate salvato il kernel dove suggerito occorrera' spostarsi nella directory **/usr/src**:

```
# cd /usr/src/
```

Estraiamo i sorgenti con:

```
root@vostropc:/usr/src/# tar xvfz linux-<VERSIONE>.tar.gz
```

Se il vostro archivio compresso e' del tipo **.tar.bz2** allora occorre sostituire l'opzione **'z'** con la opzione **'j'**.
Vedrete sfilare un elenco di file, segno che tar sta scompattando l'archivio, dovreste quindi ottenere la directory di nome **/usr/src/linux-<VERSIONE>**

Estrazione del kernel - 2/2

E' bene controllare che dentro la directory **/usr/src/** ci sia un link simbolico alla directory del kernel attuale di nome linux.

root@vostropc:/usr/src/# ls -l linux
lrwxrwxrwx 1 root src ..... linux -> linux-<VERSIONE>

Se non e' cosi' e' bene provvedere utilizzando il comando **ln**:

root@vostropc:/usr/src/# ln -s linux /usr/src/linux-<VERSION>

Applicare una patch

Bene, adesso che abbiamo estratto il nostro kernel potremmo voler applicare delle patch. Le patch sono letteralmente delle toppe che si applicano ad un software per dotarlo di opportune funzionalita'. A meno di casi particolari un utente normale non ha quasi mai la necessita' di applicare delle patch ad un kernel. Esse sono distribuite sotto forma di file di testo (generalmente .diff).
Per installare una patch e' necessario di solito salvarla in /usr/src, spostarsi su tale cartella e lanciare poi il comando:

```
root@vostropc:/usr/src/# patch -p0 < nomefilepatch.diff
```

l'opzione -p e' utilizzata per eliminare un certo numero di prefissi separati da / da ogni file che si deve patchare.

Configurazione - 1/8

A questo punto occorre configurare il nostro kernel per le nostre esigenze. In questo ci viene in aiuto una interfaccia testuale che possiamo richiamare spostandoci nella directory dei sorgenti e dando il comando:

root@vostropc:/usr/src/linux-<VERSIONE># make <tipo_interfaccia>

Per vedere i tipi di interfaccia usate **make help.**

Prima di procedere con la configurazione vera e propria e' necessario fare delle distinzioni fra funzionalita' **built-in** e **modulari.**

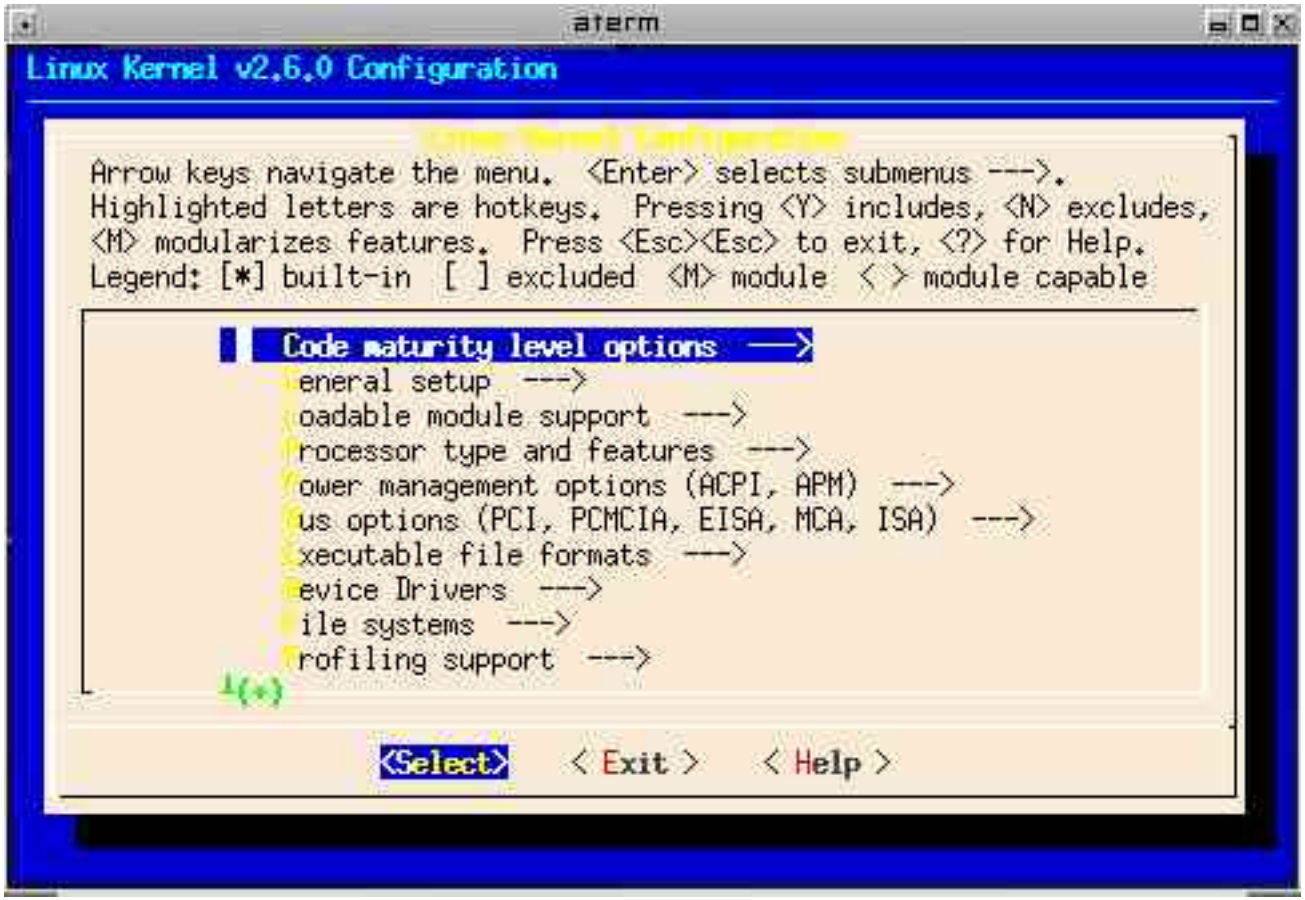
Configurazione - 2/8

• **Built-in:** vuol dire che la funzionalita' relativa e' “incorporata” all'interno del binario del kernel. Questo tipo di compilazione causa un aumento delle dimensioni finali della immagine. Pertanto e' consigliata per funzionalita' chiave del sistema (es. Supporto al fs di root, scheda di rete, etc.).

• **Modulare:** viene generato un file **nomemodulo.ko** che risiede normalmente nella cartella **/lib/modules/<versione-kernel>**. I moduli sono comodi perche' possono essere agganciati/sganciati a richiesta quando una periferica o una particolare funzionalita' e' richiesta. Vedremo come.

Vediamo le interfaccie utilizzabili per la configurazione:

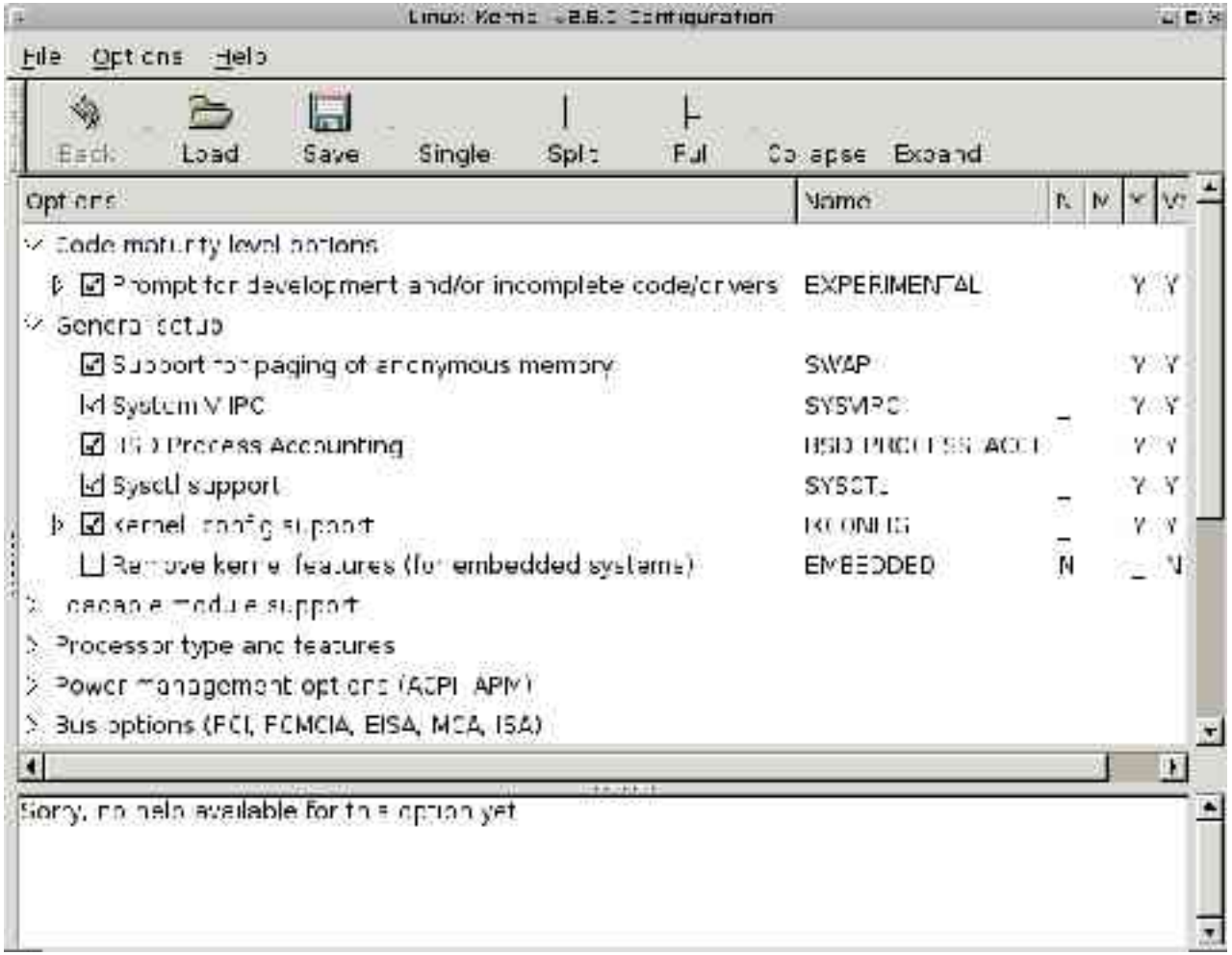
Configurazione - 3/8



Interfaccia consolle: si ottiene con il comando

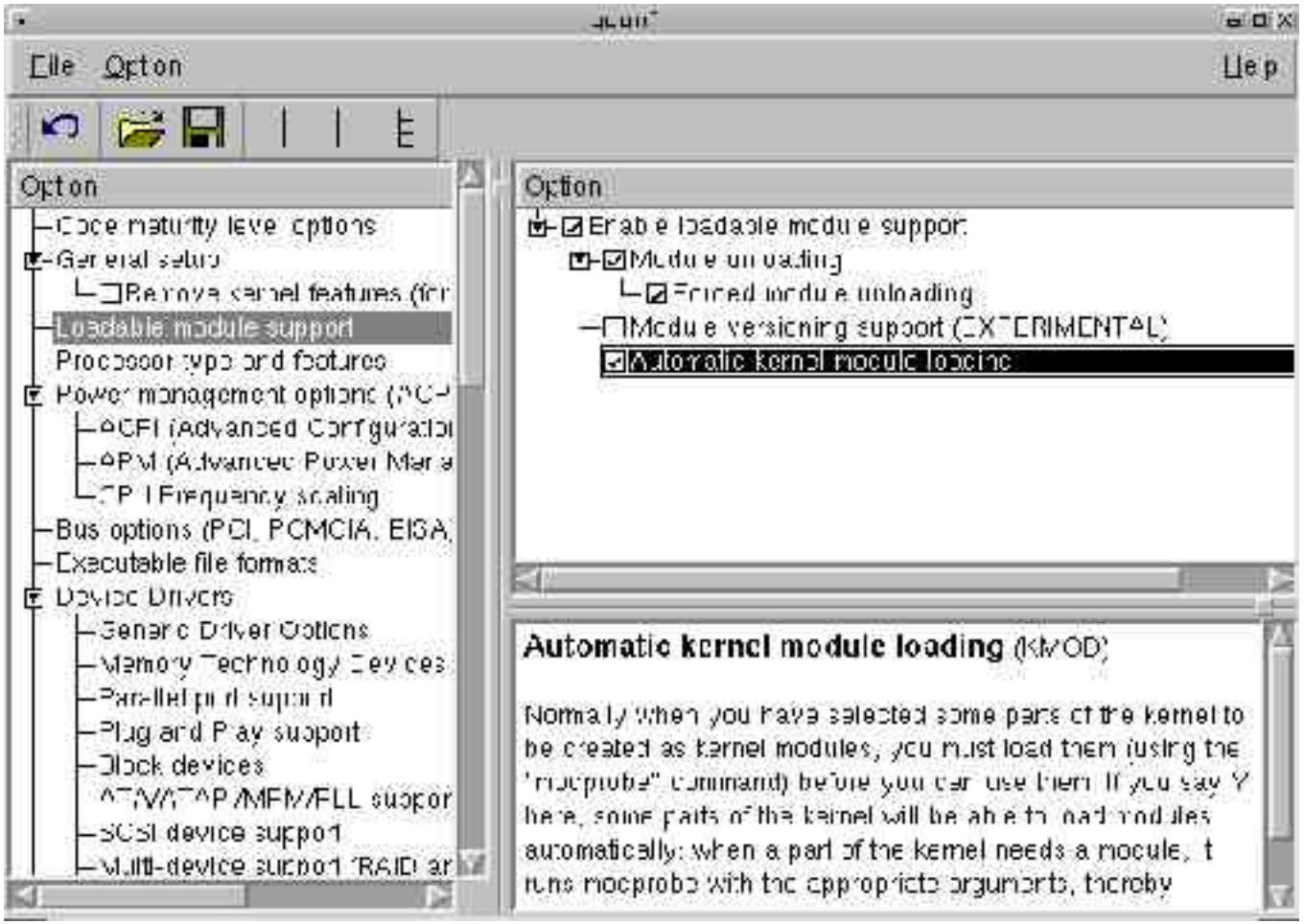
make menuconfig

Configurazione - 4/8



Interfaccia GTK: si ottiene con il comando **make gconfig**

Configurazione - 5/8



Interfaccia QT: si ottiene con il comando **make xconfig**

Configurazione - 6/8

Old style (domanda-risposta): si ottiene con il comando **make config**

Configurazione - 7/8

Al termine della configurazione si deve confermare il salvataggio delle opzioni selezionate. Rispondendo affermativamente la interfaccia produrra' un file di nome

/usr/src/linux-<VERSIONE>/.config

Questo file contiene tutte le flag selezionate, e sara' utilizzato dai comandi successivi per la compilazione effettiva del kernel. Si consiglia pertanto di farsene una copia da qualche altra parte nel proprio file system.

Ci sono alcune cose alla quale dobbiamo riservare una certa attenzione quando compiliamo un kernel, vediamo quali sono...

Configurazione - 8/8

Occore fare attenzione almeno alle voci sottostanti durante la configurazione, o potreste ottenere un kernel non proprio funzionante...

• **Selezionare il tipo di processore corettamente:** fate particolarmente attenzione a selezionare la giusta famiglia di processore per il vostro sistema.

• **Compilare il supporto per il filesystem della partizione di root come built-in:** la prima cosa che fa un kernel appena ha finito la fase di boot e' montare il file system di root (/), se il vostro kernel non ha il supporto per questo filesystem compilato built-in non riuscira' a montarlo, perche' non potra' accedere alla directory dei moduli, e andra' in panic.

Compilazione

Lanciamo la compilazione e prendiamoci un caffe':

```
root@vostropc:/usr/src/linux-<VERSIONE># make
```

questo comando compila il kernel ed i moduli.
Nel caso stiate compilando un kernel della serie 2.4 o 2.2, il comando compilera' **solo** il kernel, mentre per compilare i moduli dovrete dare il comando:

```
root@vostropc:/usr/src/linux-<VERSIONE># make modules
```

In genere a questo stadio e' difficile ricevere erorri, a meno che non abbiate fatto confusione in fase di configurazione, in tal caso controllate le scelte fatte nel menu' di configurazione.

Installazione 1/4

Copiamo la immagine del kernel nella directory **/boot**

```
root@vostropc:/usr/src/linux-<VERSIONE>#
cp arch/i386/boot/bzImage /boot/vmlinuz-<versione>
```

Per installare i moduli nel sistema occorrera' lanciare:

```
root@vostropc:/usr/src/linux-<VERSIONE>#
make modules_install
```

Il sistema iniziera' a copiare i file ***.ko** nella direcotry **/lib/modules/<versione-kernel>**, ed effettuera' altre operazioni (come chiamare depmod per generare file necessari ai programmi che gestiscono i moduli).

Installazione 2/4

Adesso che kernel e moduli si trovano nei posti giusti dobbiamo informare il Li.Lo. (Linux Loader) della presenza di un nuovo kernel. LILO e' il primo programma lanciato dal BIOS del computer. Questo file si installa nei primi settori del disco fisso, e offre un simpatico menu' dalla quale poter scegliere il kernel o il sistema operativo che vogliamo usare. Per istruire lilo riguardo ai kernel presenti su una macchina occorre editare il file **/etc/lilo.conf** con il vostro editor preferito, ed inserire/modificare le seguenti direttive:

boot=/dev/hda <- indica il disco/partizione in cui installare lilo
root=/dev/hdaX <- indica la partizione root
image=/boot/vmlinuz-<VERSIONE>
label=<vostra stringa>

Installazione 3/4

Leggete i commenti del vostro file **/etc/lilo.conf**, scoprirete che e' possibile configurare piu' di un kernel, che, su una base di N kernel e' possibile scegliere quello di default, che e' possibile passare opzioni al momento del boot al vostro kernel (es. configurazione del framebuffer, o della emulazione scsi per i masterizzatori ide).
Completate il tutto lanciando il comando:

```
# root@vostropc# lilo
```

Se il comando non si rivela particolarmente maleducato dovrebbe essere tutto a posto, riavviate la vostra macchina, al boot dovrebbe comparire un menu' e il vostro nuovo kernel sara' nella lista.

Installazione 4/4

Attenzione!

Non sovrascrivete **MAI** nel lilo.conf un kernel funzionante con uno appena compilato! Se per disgrazia il vostro nuovo kernel non dovesse partire riguadagnare il controllo di sistema sara' difficile (per un novizio).

Pertanto consiglio sempre di aggiungere una voce su lilo.conf, in modo da avere sempre una seconda alternativa da scegliere in caso di disgrazia.

Testate per bene il vostor nuovo kernel, e, solo dopo aver visto che e' tutto ok, eliminate, sempre che lo vogliate, il vecchio.

I device driver sotto Linux

I device driver sotto Linux non sono altro che moduli del kernel, depositati dentro la directory **/lib/modules/<VERSIONE-KERNEL>**

I comandi che ogni utente deve conoscere per trattare i moduli sono i seguenti:

- **INSMOD:** carica un modulo;
- **RMMOD:** rimuove un modulo;
- **LSMOD:** elenca i moduli caricati e le dipendenze fra di essi;
- **MODPROBE:** e' una utility che permette di caricare e scaricare un modulo piu' facilmente ed effettuando dei controlli in piu'.