

Gestion de Parc : Installation/Restauration par clé USB/Boot PXE

Y. Morère

Résumé

Cet article rend compte (pour l'instant) de l'installation d'une distribution Debian Sarge ainsi que de la création d'une minidistribution de sauvetage sur des machines récentes qui peuvent booter sur une clé USB.

Table des matières

1	Introduction	2
2	Installation de la distribution via clé USB	2
2.1	Préparation de la clé (Version Quick & Dirty)	2
2.2	Intermède réparation de système	3
2.3	Utilisation de grub pour dépannage <i>bootloader</i>	4
2.4	Reprise	5
2.5	Préparation de la clé (Version Mr. Proper)	6
2.6	Sélection du périphérique de Boot	8
2.7	Démarrage de l'installation	8
3	Création d'une MiniDistrib de Restauration	8
3.1	Prérequis	8
3.2	Création d'un noyau sans modules	9
3.3	Création du système de fichier	12
3.3.1	Utilisation de <i>buildroot</i>	12
3.3.2	Configuration de notre distribution	51
3.4	Configuration de SYSLinux et test	54
3.4.1	Explications des différentes parties	54
3.4.2	Bonus : modification de l'image <i>splash.rle</i>	56
3.4.3	Tests	57
4	Création et utilisation de la clé USB	60
4.1	Création de la clé	60
4.2	Clonage de machine	62
5	Crédits	63

1 Introduction

Les dernières séries de machines achetées par notre université, pour des raisons de coût et de sécurité, ne disposent pas de lecteur de disquette et de lecteur CD-Rom.

Toutes ces machines sont basées sur une des cartes mères Intel (chipset Intel i915, réseau son et vidéo intégrés), un disque dur SATA 80Go.

Afin de pouvoir y installer facilement et rapidement Linux (dans un premier temps) deux solutions s'offre à nous.

1. Booter sur une clé USB qui contient l'installateur de la distribution choisie, puis réaliser l'installation une fois le réseau configuré ;
2. posséder un serveur TFTP configuré sur une autre machine Unix, et booter via le réseau et le protocole PXE (PXE = Preboot Execution Environment. C'est un protocole implémenté dans la BootRom des cartes Ethernet, sollicité par un boot réseau, pour interroger un serveur.) afin de charger une image noyau et système de fichier qui lancera l'installateur.

Dans une premier temps nous analyserons le boot sur la clé USB.

2 Installation de la distribution via clé USB

Il y a pas mal de documentation sur ce sujet. J'ai basé la petite expérience sur les informations du manuel d'installation Debian.

2.1 Préparation de la clé (Version Quick & Dirty)

Dans un premier temps il est nécessaire de récupérer différents fichiers. Pour cela l'adresse <ftp://cdimage.debian.org/> est un bon point de départ.

Voici la liste des fichiers à récupérer :

- ▷ <ftp://cdimage.debian.org/debian-cd/current/i386/iso-cd/debian-31r0a-i386-netinst.iso> ou <ftp://cdimage.debian.org/debian-cd/current/i386/iso-cd/debian-31r0a-i386-businesscard.iso> : une image iso qui tiendra sur votre clé USB (pour ma part j'ai utilisé une clé 64Mo, j'ai donc choisi la version *businesscard*).
- ▷ <ftp://cdimage.debian.org/debian/dists/stable/main/installer-i386/current/images/hd-media/boot.img.gz> : c'est l'image bootable que l'on va écrire sur la clé USB. Elle contient les noyaux 2.4 et 2.6 ainsi que les *initrd* correspondants, un système de fichiers qui sera chargé en RAM.

Voilà, tout ce petit monde téléchargé, on va pouvoir commencer.

Je suivrai ce qui est indiqué à l'adresse suivante : <http://www.debian.org/releases/stable/i386/ch04s04.html.fr>

Je pars bien sur du principe que vous utilisez une distribution Linux. Il est possible de réaliser la même chose sous Windows à l'aide de *RawWrite for windows* <http://uranus.it.swin.edu.au/~jn/linux/rawwrite.htm>.

Dixit l'adresse <http://www.debian.org/releases/stable/i386/ch04s04.html.fr>, paragraphe 4.4.1. méthode souple

Le fichier `hd-media/boot.img.gz` contient tous les fichiers de l'installateur, le noyau et le programme SYSLINUX avec son fichier de configuration. Il vous suffit d'extraire ces fichiers dans votre clé USB :

```
# zcat boot.img.gz > /dev/sda
```

Bien sûr toutes les données présentes sur la clé seront détruites. Utilisez le bon périphérique pour votre clé!

SYSLINUX est un *boot loader* pour le système d'exploitation Linux qui utilise un system de fichier FAT MSDOS/Windows. Il doit permettre de simplifier la première installation de Linux et la création de disque de récupération.

Bien sur comme l'image Debian est prévue pour une clé de 128Mo, le système croira qu'elle fait 128Mo, alors qu'elle n'en fait que 64. Il faut donc faire bien attention de ne pas dépasser 64Mo de donnée sur la clé, sous peine d'obtenir quelque chose de non utilisable (Dirty).

2.2 Intermède réparation de système

Je peux vous dire que la dernière phrase n'est pas innocente (Utilisez le bon périphérique pour votre clé!). Lors de ma première tentative, j'ai eu la malheureuse expérience de taper `hda`, à la place de `sda`. Vous imaginez facilement les dégâts occasionnés...

Mon premier disque de 60Go (partition `swap root` et `home`) a été écrasé avec une image de 128Mo formaté en fat16 et la table de partition complètement erronée.

A ma grande surprise, le système fonctionnait toujours, et semblait stable. En fait j'ai eu pas mal de chance. En effet, lors de l'installation de ma Linuxbox, j'ai placé le swap (500Mo) en tout début de disque (première partition). L'écriture de l'image a donc écrasé des données du swap et la table des partitions, mais aucune donnée de `/home` et aucun programme de `/`.

C'est à ce moment précis, qu'il faut prendre tout son temps et ne rien précipiter. La table de partition étant complètement erronée, le moindre redémarrage rendrait la machine inutilisable.

Dans une premier temps j'ai donc sauvegarder mes données sur une autre machine.

Puis j'ai consulté le fichier `/proc/partitions` afin d'avoir des infos sur les partitions. En effet il est nécessaire de les recréer comme à l'origine, sinon impossible de redémarrer.

```
yann@tuxpowered:~/ more /proc/partitions
```

```
major minor #blocks name
3      0  58615704 hda
3      1  37889271 hda1
3      2           1 hda2
3      5  20723818 hda5
3     64  19551168 hdb
3     65   9767488 hdb1
3     66   289170 hdb2
3     67   9494415 hdb3
```

```
yann@tuxpowered:~/ $
```

`fdisk` ou `cdisk` ont besoin des numéros de secteurs, ou taille en Ko ou Mo pour recréer les partitions. Mais en analysant les réponses des commandes `df` avec différentes options et `more /proc/partitions` je n'ai pas pu déterminer de manière sure la taille en Mo de chaque partition. Les nombres renvoyés ne correspondaient pas exactement.

Il fallait donc passer par les nombres d'unité de `fdisk` ou `cdisk`

Je me suis tourné vers `testdisk` <http://www.cgsecurity.org/index.html?testdisk.html>, qui se trouve dans les paquets `debian`.

Ce dernier a détecté mes partitions `/` et `/home` ainsi que le numéro d'unité de démarrage et de fin.

J'avais donc toutes les informations pour recréer les partitions à l'initial. Un `fdisk` plus tard je redemarre la machine, avec une certaine anxiété.

Cette dernière ne boote pas (curseur qui clignotte en haut à gauche). Pas de panique, avec l'écriture de la table de partition, et de l'image, Grub a disparu.

2.3 Utilisation de grub pour dépannage *bootloader*

Il faut donc créé une disquette de boot Grub, qui va permettre de réinstaller le *bootloader*.

L'adresse suivante donne toutes les informations pour réaliser cela http://doc.drimm.u-bordeaux1.fr/documentation/linux/disquette_grub ou encore celle-ci http://www.finix.eu.org/spip/article.php3?id_article=69

On considère que la construction se fait depuis une machine Linux avec grub déjà installé.

Création d'un système de fichier linux :

```
mkfs.ext2 /dev/fd0
```

Copie des fichiers de grub sur la disquette :

```
# on suppose que le répertoire /mnt/floppy existe
mount /dev/fd0 /mnt/floppy/
mkdir /mnt/floppy/boot/
cp -rp /boot/grub /mnt/floppy/boot/

# ATTENTION a ne pas oublier de démonter la disquette
# sinon tous les fichiers ne seront pas dessus...
umount /mnt
```

On va rendre la disquette bootable. Cela se fait avec le shell grub qui offre les mêmes fonctionnalités en ligne de commande (système linux déjà démarré) que lors du boot. Invoquer grub sous linux :

```
grub

# maintenant on est dans le shell grub et
# le prompt grub> apparait
grub>

grub> root (fd0)
grub> setup (fd0)

# sortir de grub
grub> quit
```

Voilà c'est fini, il ne reste plus qu'à booter sur la disquette et donner les commandes qui vont permettre d'installer grub sur le disque dur. Il faut commencer par lancer *grub*

```
grub
# maintenant on est dans le shell grub et
# le prompt grub> apparait
grub>install (hd0,1)/boot/grub/stage1 (hd0) (hd0,1)/boot/grub/stage2 \
(hd0,1)/boot/grub/menu.lst
# dans mon cas tout est encore sur le disque dur, j'utilise les fichiers grub du disque
```

Il est aussi possible, à partir d'une disquette grub, de booter un système linux, grâce aux fichiers qui se trouve dans ses partitions.

```
grub
# maintenant on est dans le shell grub et
# le prompt grub> apparait
grub>kernel (hd0,1)/boot/vmlinuz-2.6.8-2-686 root=/dev/hda2
#on indique quel noyau charger avec les options
grub>initrd (hd0,1)/boot/initrd-2.6.8-2-686
#on indique quelle image ramdisk
grub>boot
#on boot
# on peut remarque qu'un appui sur TAB permet la complétion automatique comme bash
```

Voilà le système est de nouveau opérationnel, on peut poursuivre...

2.4 Reprise

Le fichier `hd-media/boot.img.gz` contient tous les fichiers de l'installateur, le noyau et le programme SYSLINUX avec son fichier de configuration. Il vous suffit d'extraire ces fichiers dans votre clé USB :

```
# zcat boot.img.gz > /dev/sda
```

Bien sûr toutes les données présentes sur la clé seront détruites. Utilisez le bon périphérique pour votre clé !

Cette fois ci, tout s'est bien passé malgré un petit message d'erreur du à la taille de ma clé USB de 64Mo alors que le site debian nous informe que cette image est prévue pour des clés de taille supérieure ou égale à 128Mo.

Mais je voulais éviter de réaliser les opérations à la main comme décrites ici <http://www.debian.org/releases/stable/i386/ch04s04.html.fr>

Montez ensuite la clé (`mount /dev/sda /mnt`), qui aura maintenant un système de fichiers FAT16, et copiez une image ISO de type `netinst` ou `businesscard`. Le nom de ce fichier doit se terminer par `.iso`. Démontez la clé (`umount /mnt`) et voilà, c'est fait !

Pour ma part j'ai donc copier l'image iso *businesscard*. j'obtiens finalement ceci :

```
tuxpowered:~/# mount /dev/sda /mnt
tuxpowered:~/# ls /mnt/
debian-31r0a-i386-businesscard.iso  f4.txt  initrd26.gz  splash.rle
disk.lbl                          f5.txt  initrd.gz    syslinux.cfg
f10.txt                            f6.txt  initrd.list  syslinux.txt
f1.txt                             f7.txt  ldlinux.sys
f2.txt                             f8.txt  linux
f3.txt                             f9.txt  linux26
tuxpowered:~/# df
Sys. de fich.      1K-blocs      Occupé Disponible Capacité Monté sur
[...]
/dev/sda           125688        38988      86700  32% /mnt
tuxpowered:~/#
```

Notre clé USB est maintenant prête pour réaliser l'installation.

2.5 Préparation de la clé (Version Mr. Proper)

Comme la taille de la clé est inférieure à l'image fournie par Debian, on va suivre ce qui est indiqué à l'adresse suivante <http://www.debian.org/releases/stable/i386/ch04s04.html.fr>.

4.4.2. Copie des fichiers la méthode souple

Si vous aimez la souplesse ou si vous voulez simplement savoir ce qui se passe, vous pouvez utiliser la méthode suivante pour mettre les fichiers sur la clé.

4.4.2.1. Partitionner un périphérique USB sur Intel x86

Nous montrerons comment n'utiliser que la première partition, au lieu de tout le périphérique. Note

La plupart des clés USB sont préconfigurées avec une seule partition FAT16. Vous n'aurez sans doute pas à partitionner ou à formater la clé. Si vous devez le faire, utilisez le programme `fdisk` ou un autre programme de partitionnement pour créer cette partition et tapez :

```
# mkdosfs /dev/sda1
```

Faites attention à utiliser le périphérique correct pour la clé. Le programme `mkdosfs` se trouve dans le paquet Debian `dosfstools`.

Pour pouvoir lancer le noyau après l'amorçage sur la clé, nous y mettons un programme d'amorçage. Tous les programmes d'amorçage fonctionnent (p. ex. LILO), mais SYSLINUX est préférable car il utilise une partition FAT16 et peut être reconfiguré en modifiant un simple fichier texte. On peut modifier la configuration du programme d'amorçage avec tous les systèmes d'exploitation qui acceptent le système de fichiers FAT.

Pour mettre SYSLINUX sur la partition FAT16 de la clé, installez les paquets `syslinux` et `mtools` sur votre système et faites :

```
# syslinux /dev/sda1
```

Encore une fois, faites bien attention à utiliser le nom correct pour la clé. La partition ne doit pas être montée au lancement de SYSLINUX. Cette procédure écrit un secteur d'amorçage sur la partition et crée le fichier `ldlinux.sys` qui contient le code du programme d'amorçage.

Montez la partition (`mount /dev/sda1 /mnt`) et copiez les fichiers suivants sur la clé :

- ▷ `vmlinuz` (noyau binaire)
- ▷ `initrd.gz` (image du disque virtuel initial)
- ▷ `syslinux.cfg` (fichier de configuration de SYSLINUX)
- ▷ Modules optionnels du noyau

Si vous voulez modifier le nom des fichiers, remarquez que SYSLINUX ne peut traiter que des noms de type DOS (8.3).

Le fichier de configuration `syslinux.cfg` doit contenir les deux lignes suivantes :

```
default vmlinuz
append initrd=initrd.gz ramdisk_size=12000 root=/dev/rd/0 init=/linuxrc rw
```

Remarquez que la valeur du paramètre `ramdisk_size` doit être augmentée avec la taille de l'image qui est amorcée. En cas d'échec, vous pouvez ajouter `devfs=mount,dall` à la ligne « `append` »

4.4.2.2. Ajouter une image ISO

Vous pouvez maintenant mettre une image ISO (`businesscard`, `netinst` ou même l'image complète) sur la clé. Le nom de cette image doit se terminer par `.iso`.

Si vous voulez faire une installation sur le réseau, sans utiliser d'image ISO, vous sauterez bien sûr l'étape précédente. De plus vous devrez utiliser le disque virtuel initial qui se trouve dans le répertoire netboot au lieu de celui dans le répertoire hd-media, car le fichier hd-media/initrd.gz n'a aucun support pour le réseau.

Quand tout est fait, démontez la clé USB (umount /mnt) et activez la protection contre l'écriture.

4.4.2.3. Amorcer la clé USB

Avertissement

Si le système refuse de s'amorcer à partir de la clé, il se peut que la clé possède un secteur principal d'amorçage défectueux. Corrigez-le avec le programme `install-mbr` qui se trouve dans le paquet `mbr` :

```
# install-mbr /dev/sda
```

Voilà en suivant tout cela on doit obtenir une clé usb qui boote. Sauf que cela ne marche pas du tout chez moi.

Quelle que soit la configuration du bios de ma carte Epia (USB-ZIP ou USB-HDD) rien n'y fait, la clé n'est pas reconnue comme bootable.

Ne sachant pas si cela vient de la clé USB, ou encore du bios de la carte mère (pour cela voir <http://syslinux.zytor.com/usbkey.php>), et en remarquant que l'image Debian utilise la méthode «superfloppy» (méthode qui consiste à utiliser la totalité du média sans table de partition), je décide d'employer une autre méthode.

Il y a deux manières d'accéder aux disques : la méthode ordinaire consiste à créer une ou plusieurs partitions, la seconde consiste à accéder au disque en mode brut, on appelle cela le format «superfloppy» dans les environnements Dos/Windows.

Voir aussi <http://www.marlow.dk/site.php/tech/usbkeys> pour formater la clé comme un disque dur.

Pour l'instant voici une méthode qui permet de booter la clé USB, mais il n'est pas possible de monter le contenu de la clé dans un répertoire. On verra que l'installation de SYSLINUX corrompt la table de partition.

Bien sûr ici tout se fait root, il faut commencer par connecter la clé.

```
yoda:/home/yann/temp/sarge/hd-media# mkdosfs -I /dev/sda
#pour forcer toute la clé en dosfs
yoda:/home/yann/temp/sarge/hd-media# mkdir ramdisk
yoda:/home/yann/temp/sarge/hd-media# syslinux /dev/sda
#rendre amorçable la clé
yoda:/home/yann/temp/sarge/hd-media# mount /dev/sda ramdisk/
#monter le volume
yoda:/home/yann/temp/sarge/hd-media# gunzip boot.img.gz
yoda:/home/yann/temp/sarge/hd-media# mkdir image
yoda:/home/yann/temp/sarge/hd-media# mount -o loop boot.img image/
# monter en dans un répertoire l'image
yoda:/home/yann/temp/sarge/hd-media# ls image/
disk.lbl  f2.txt  f5.txt  f8.txt      initrd.gz  linux      syslinux.cfg
f10.txt  f3.txt  f6.txt  f9.txt      initrd.list linux26    syslinux.txt
f1.txt   f4.txt  f7.txt  initrd26.gz ldlinux.sys splash.rle
yoda:/home/yann/temp/sarge/hd-media# cp -r image/* ramdisk/
# tout y copier
yoda:/home/yann/temp/sarge/hd-media# umount ramdisk/
yoda:/home/yann/temp/sarge/hd-media# umount image/
```


Maintenant votre clé est bootable, et il est possible de démarrer l'installation d'un système debian.

2.6 Sélection du périphérique de Boot

Dans un premier temps, il est nécessaire de configurer le bios de votre machine afin qu'elle utilise comme premier périphérique de démarrage la clé USB. Il faut bien sur que cette option soit disponible. Je vous renvoie donc à la documentation de votre carte mère pour plus d'information.

En ce qui concerne les cartes mère Intel à base i915, la configuration se fait en deux temps.

Il faut tout d'abord, vérifier que votre clé usb est détectée dans le bios, suivant les cartes, soit comme un disque dur, soit comme un périphérique amovible. Pour cela je vous conseille de brancher la clé avant de démarrer à froid la machine.

Dans la liste ou apparaît votre clé, mettre la clé comme premier périphérique (sinon elle n'apparaîtra pas dans la liste des périphériques bootables).

Dans un second temps, il faut choisir l'ordre des périphériques de boot, vous placerez bien sur la clé en premier.

Voilà, il ne reste plus qu'à sauvegarder les modifications et rebooter.

2.7 Démarrage de l'installation

La machine reboote, et un splash-screen debian apparaît avec l'invite d'installation habituelle. Je vous conseille de saisir `linux26` et `enter` afin de booter avec un noyau 2.6. Dans le cas contraire c'est un noyau 2.4 qui démarrera et il sera incapable de trouver vos disques SATA.

La suite reste identique à une installation classique, et je n'irai donc pas plus loin.

L'installation s'est bien déroulée. La version de Xfree86 ne gère pas le chipset i915 et c'est la module ves a qui a été choisi. La carte réseau est elle aussi bien reconnue (l'installation s'est fait par le réseau). Seule la carte son n'a pas été testée.

3 Création d'une MiniDistrib de Restauration

La suite du document s'inspire très largement de la méthode utilisée pour la création de la disquette BSC développée au CRIUM <http://www.crium.univ-metz.fr/> que l'on peut trouver à l'adresse suivante <http://www.crium.univ-metz.fr/docs/parc/bsc/>.

3.1 Prérequis

Notre petite distribution doit au final être capable de réaliser une réparation et le cas échéant une restauration des partitions par réseau en bootant de la clé USB. Pour cela il est nécessaire que le matériel soit reconnu (disque SATA) et que le réseau soit opérationnel (carte réseau et connexion au serveur Unix qui contient les images des partitions).

La solution choisie est la création d'un gros noyau monolithique sans modules (la taille importe peu vu que l'on a la place sur la clé) qui embarquera un maximum de pilotes de périphériques.

En ce qui concerne le système de fichier, il sera basé sur une `busybox`, et sera créé à l'aide de l'outil `buildroot`.

3.2 Création d'un noyau sans modules

Le noyau est un 2.6.8 livré avec la debian sarge. Il s'agit donc d'installer les sources de ce noyau, de les décompresser et de le construire avec les bonnes options.

```
# apt-get install kernel-source-2.6.8
# cd /usr/src
# tar xjf kernel-source-2.6.8.tar.bz2
# ln -s kernel-source-2.6.8 linux #pour la busybox
# cd linux
```

Il faut lancer au choix (tout dépend de vos préférences),

```
# make xconfig
# make menuconfig
# make gconfig
# make kconfig
```

Seule l'interface sera différente, au final vous obtenez un fichier `.config` qui contient la configuration du noyau à compiler.

Il y a quelques écueils à éviter. Tout d'abord ne pas activer le support `devfs` qui est maintenant obsolète. Nous allons utiliser `udev` par l'intermédiaire de la `busybox`, et il faut pour cela activer `sysfs`.

Pour le reste, ne pas oublier d'intégrer exclusivement dans le noyau (Y) et ne pas créer de module (M)

Une fois la configuration effectuée, vous pouvez vérifier que vous n'avez pas de module par la commande :

```
yoda:/usr/src/linux# grep "=m" .config
yoda:/usr/src/linux#
```

Si quelques lignes apparaissent, il faut reprendre la configuration de votre noyau.

Voici quelques captures de ma configuration :

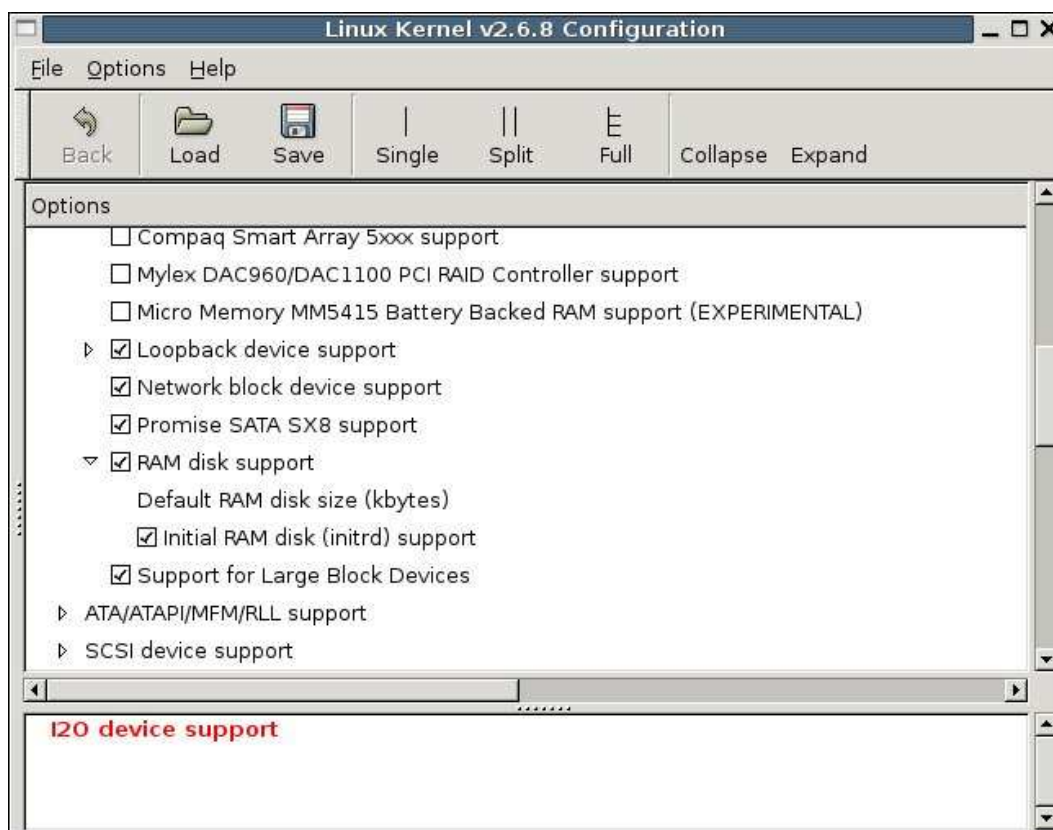


FIG. 1 – Configuration pseudo système de fichier

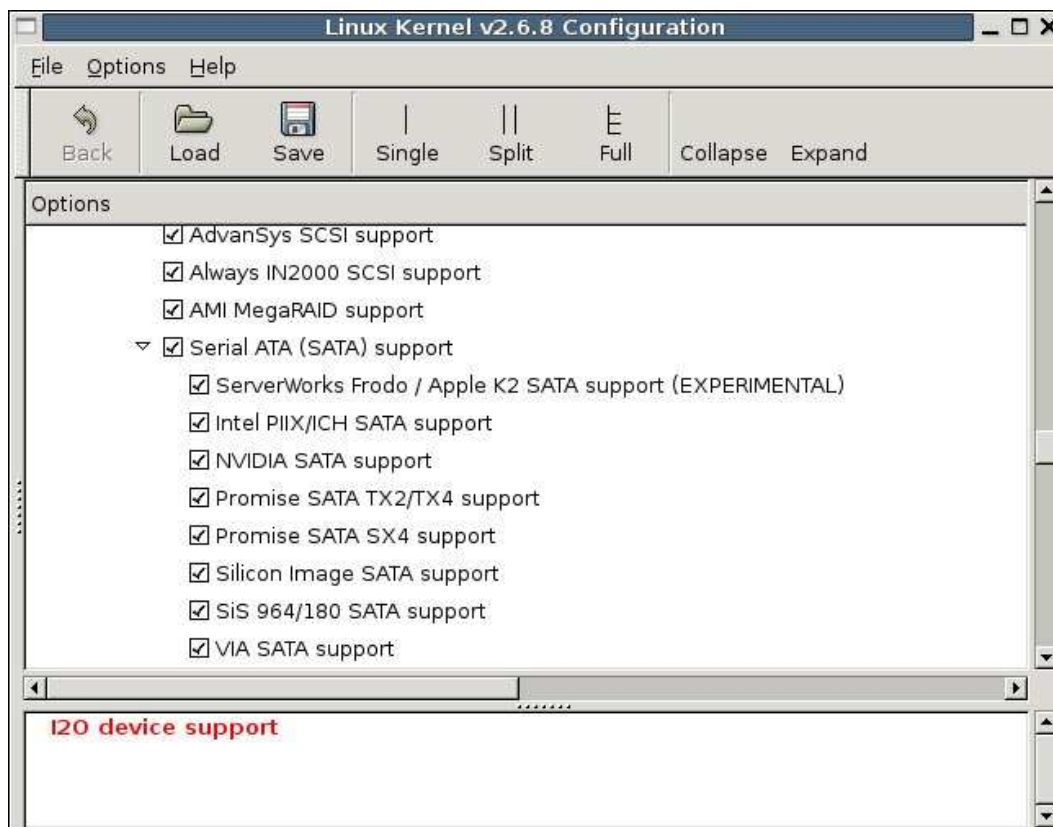


FIG. 2 – Configuration pilotes Serial ATA

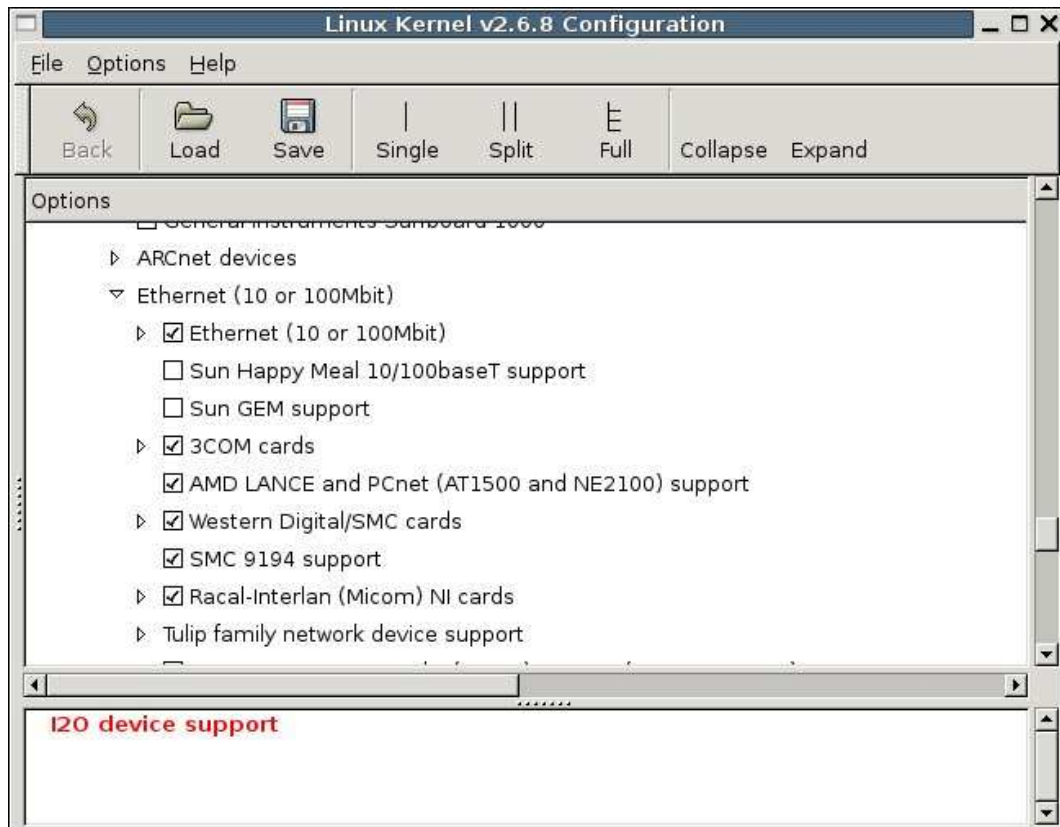


FIG. 3 – Configuration pilotes réseau

Et pour ceux qui ne veulent pas passer 1/2 heure à passer en revue toutes les options voici mon fichier de configuration (il doit être assez complet). Le noyau généré fait 3.8Mo compressé bzip2 et 9Mo décompressé.

▷ [fichier_utils/config_linuzrs2_avec_sysfs_avec_automounter](#)

▷ [fichier_utils/config_linuzrs2_avec_sysfs](#)

et voici les deux noyaux

▷ [fichier_utils/linuzrs2_avec_sysfs_avec_automounter](#)

▷ [fichier_utils/linuzrs2_avec_sysfs](#)

On peut maintenant placer la noyau sur notre clé USB. A cause d'une limitation de syslinux, il est nécessaire que le nom du noyau soit au format 8.3. C'est pour cela que les noyaux Debian sont nommés `linux` et `linux26` et non comme dans votre répertoire `boot` : `vmlinuz-2.6.8-2-686`. Il m'a été dit que les noyaux 2.6.8 avaient quelques problèmes avec l'USB, j'ai donc réitérer la compilation avec un 2.6.12.2.

On pourra par exemple le nommer `linuzrs` pour le noyau `linux` "rescue" compressé.

l'image de ce noyau se trouve ici [fichier_utils/last/vmlinuz-2.6.12.2](#) et son fichier de configuration là [fichier_utils/last/config-2.6.12.2](#).

```
yoda:/home/yann/temp/sarge/hd-media# mount /dev/sda ramdisk/
yoda:/home/yann/temp/sarge/hd-media# cp /usr/src/linux/arch/i386/boot/bzImage ramdisk/li
yoda:/home/yann/temp/sarge/hd-media# umount /dev/sda
```

On verra la configuration de SYSLINUX un peu plus tard.

3.3 Création du système de fichier

Si vous ne le savez pas, un noyau seul ne peut pas faire grand chose, il va booter et s'il ne trouve pas de système de fichier pour lancer le premier processus `init`, vous aurez droit à une joli `kernel panic`.

Il est donc nécessaire de créer un système de fichier racine (`root file system`), de plus ce système de fichier devra être placé dans la ram. En effet il faut pouvoir booter le système même s'il n'y a pas de disque.

Ce système de fichier complet devra contenir une arborescence complète Unix (`/dev`, `/usr` etc...) ainsi que différents programmes qui doivent permettre de se connecter au réseau, partitionner un disque, vérifier un disque ou une partition, cloner un disque...

De plus ce système doit être léger en taille, pour cette raison nous allons nous diriger tout droit vers la `busybox` <http://www.busybox.net/about.html> qui se définit comme le couteau Suisse du Linux embarqué (BusyBox : The Swiss Army Knife of Embedded Linux).

C'est un programme exécutable qui renferme un grand nombre de programme du monde Unix. Je vous renvoie à cette adresse pour une liste exhaustive <http://www.busybox.net/downloads/BusyBox.html>.

De plus ce programme est hautement configurable afin d'embarquer plus ou moins de programme. Il est donc possible de l'adapter à ses besoins.

Dès lors que l'on parle de système embarqué, il vient de suite l'utilisation de la micro-libc `uClibc` <http://uclibc.org/>, qui est une librairie C pour les systèmes embarqués. Elle est beaucoup plus légère que le librairie GNU C.

IL va donc falloir construire la `busybox` à l'aide de la `uClibc` puis construire le système de fichier afin de pouvoir construire une image de ce système de fichier qui puisse être chargé en mémoire.

C'est un vaste programme, qui va nous être largement facilité par l'utilisation de `buildroot` <http://buildroot.uclibc.org/>. `Buildroot` est un ensemble de "patches" et Makefiles qui rendent plus facile la compilation (cross-compilation) de la série d'outils et du système de fichier racine (root filesystem) pour le système embarqué Linux souhaité en utilisant la `uClibc`. L'intérêt de `Buildroot` est qu'il va télécharger, compiler tous les outils que vous aurez sélectionnés lors de la configuration et ensuite vous créer votre système de fichier racine.

3.3.1 Utilisation de buildroot

Nous allons tout d'abord télécharger la version journalière `buildroot-snapshot.tar.bz2` de `buildroot`. Elle est disponible à l'adresse <http://buildroot.uclibc.org/downloads/snapshots/>

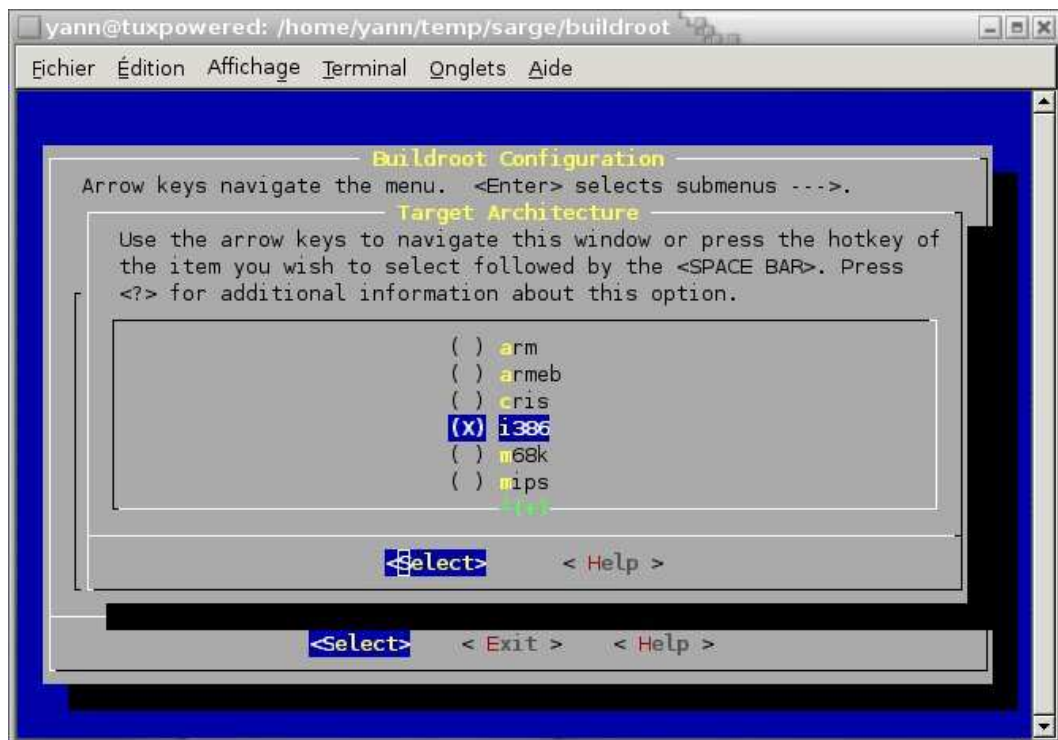
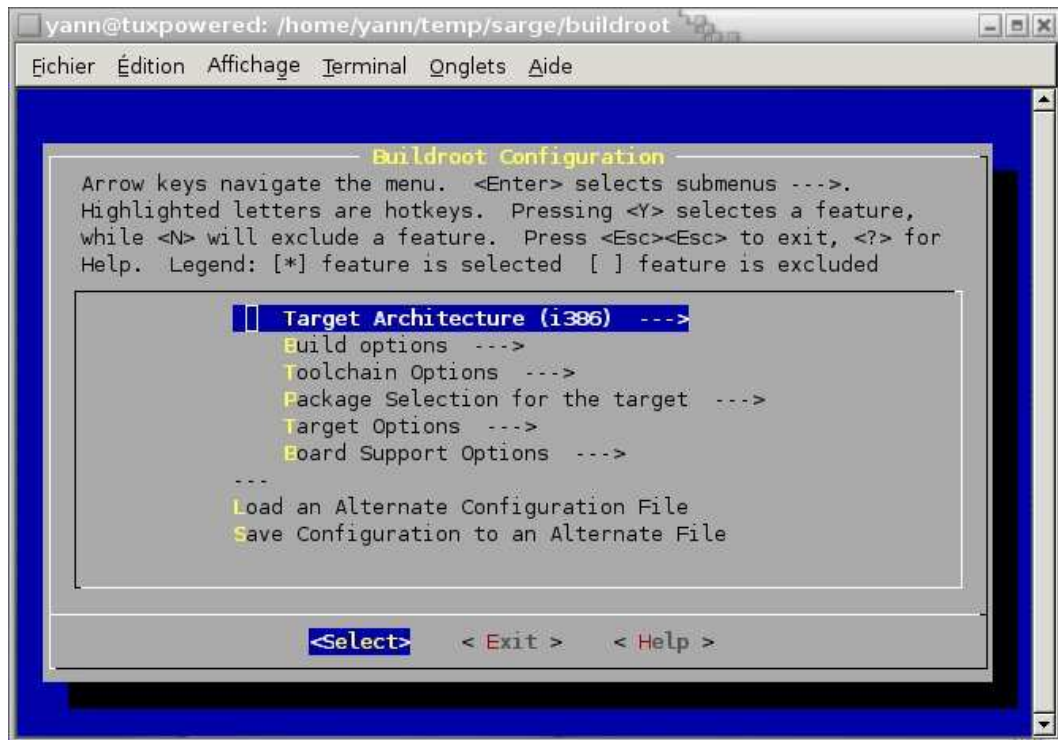
Il n'est pas nécessaire d'être `root` pour compiler une image de système de fichier. On va donc décompresser notre paquets, puis lancer la configuration :

```
yann@tuxpowered:~/temp/sarge$ tar xjf buildroot-snapshot.tar.bz2
yann@tuxpowered:~/temp/sarge$ cd buildroot
yann@tuxpowered:~/temp/sarge/buildroot$ make menuconfig
```

Voici la configuration de `buildroot` en image. Le fichier de configuration produit se trouve ici [fichier_utils/last/buildroot.config](#)

Il est à noter que j'ai fixé la taille de l'image final du système de fichier afin de pouvoir y copier d'autres fichiers.

```
(12000) size in blocks (leave at 0 for auto calculation)
```



```

yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Build options --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded

[*]get --passive-ftp) wget command
(svn co) Subversion (svn) checkout command
[ ] Tar verbose
(unc) Sourceforge mirror site
($(BUILD_DIR)/staging_dir) Toolchain and header file location?
(1) Number of jobs to run simultaneously

<Select>  < Exit >  < Help >

```

```

yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Toolchain Options --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded

[-] Kernel Header Options
  Kernel Headers (Linux 2.6.11 kernel headers) --->
  --- uClibc Options
  [*] Use the daily snapshot of uClibc?
  [*] Enable locale/gettext/i18n support?
  --- Binutils Options
  Binutils Version (binutils 2.16.1) --->
  --- Gcc Options
  GCC compiler Version (gcc 3.4.2) --->
  () Additional gcc options

<Select>  < Exit >  < Help >

```



```

yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Toolchain Options --
Arrow keys navigate the menu. <Enter> selects submenus ---.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded
+---+
[ ] Build/install c++ compiler and libstdc++?
[ ] Build/install Objective-C compiler and runtime?
--- Ccache Options
[ ] Enable ccache support?
--- Gdb Options
[ ] Build gdb debugger for the Target
[ ] Build gdb server for the Target
[ ] Build gdb client for the Host
-- Common Toolchain Options
[*] Enable multilib support?
+---+

<Select>  < Exit >  < Help >

```

```

yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Toolchain Options --
Arrow keys navigate the menu. <Enter> selects submenus ---.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded
+---+
--- Ccache Options
[ ] Enable ccache support?
--- Gdb Options
[ ] Build gdb debugger for the Target
[ ] Build gdb server for the Target
[ ] Build gdb client for the Host
--- Common Toolchain Options
[*] Enable multilib support?
[*] Enable large file (files > 2 GB) support?
-- (-Os -pipe) Target Optimizations
+---+

<Select>  < Exit >  < Help >

```



```

yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Package Selection for the target --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded

-- The default minimal system
[*] BusyBox
[*] Use the daily snapshot of BusyBox?
(package/busybox/busybox.config) BusyBox configuration file to use?
--- The minimum needed to build a uClibc development system
[ ] bash
[ ] bzip2
[*] coreutils
[ ] diffutils
[ ] ed
<Esc>

<Select>  < Exit >  < Help >

```

```

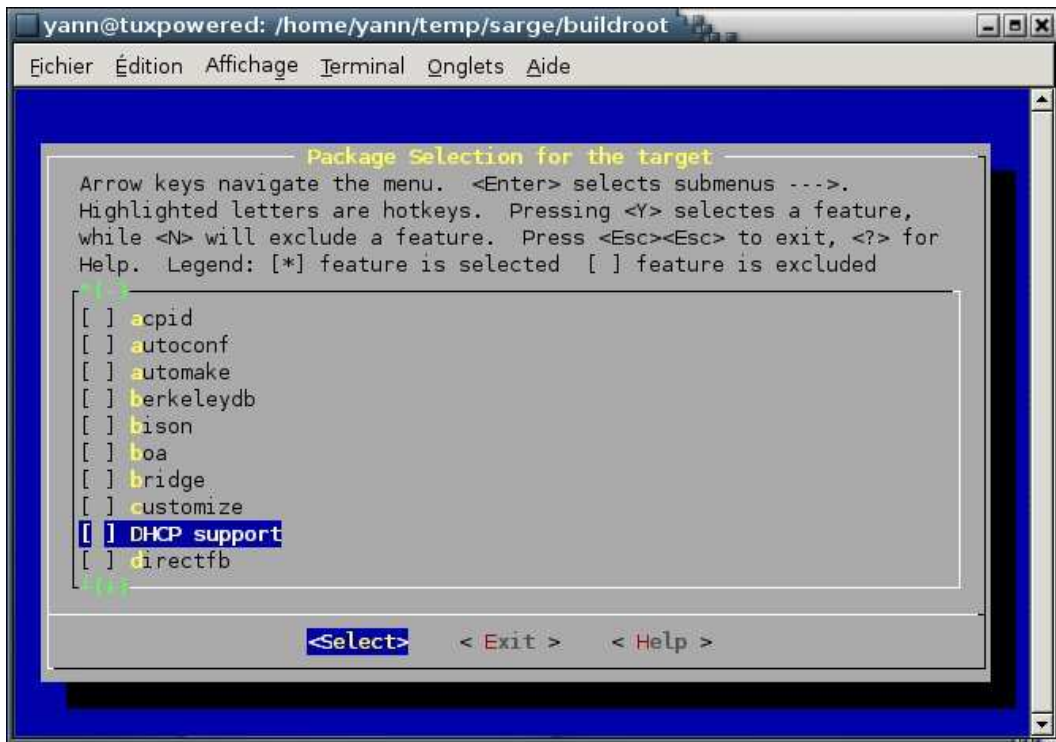
yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Package Selection for the target --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded

<Esc>
[ ] findutils
[ ] flex
[ ] gawk
[ ] native toolchain in the target filesystem
[ ] ccache support in the target filesystem
[ ] grep
[ ] make
[ ] patch
[[ ] sed
[ ] tar
<Esc>

<Select>  < Exit >  < Help >

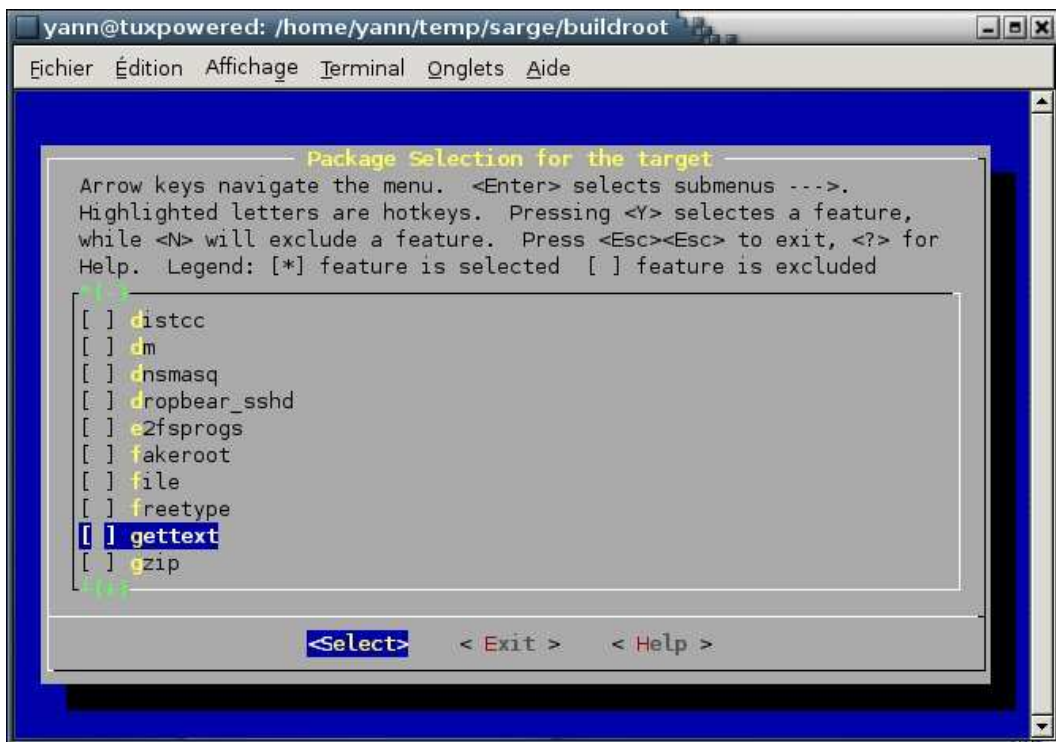
```



```
yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Package Selection for the target --
Arrow keys navigate the menu.  <Enter> selects submenus --->.
Highlighted letters are hotkeys.  Pressing <Y> selects a feature,
while <N> will exclude a feature.  Press <Esc><Esc> to exit, <?> for
Help.  Legend: [*] feature is selected [ ] feature is excluded
^~^~
[ ] acpid
[ ] autoconf
[ ] automake
[ ] berkeleydb
[ ] bison
[ ] boa
[ ] bridge
[ ] customize
[*] DHCP support
[ ] directfb
^~^~

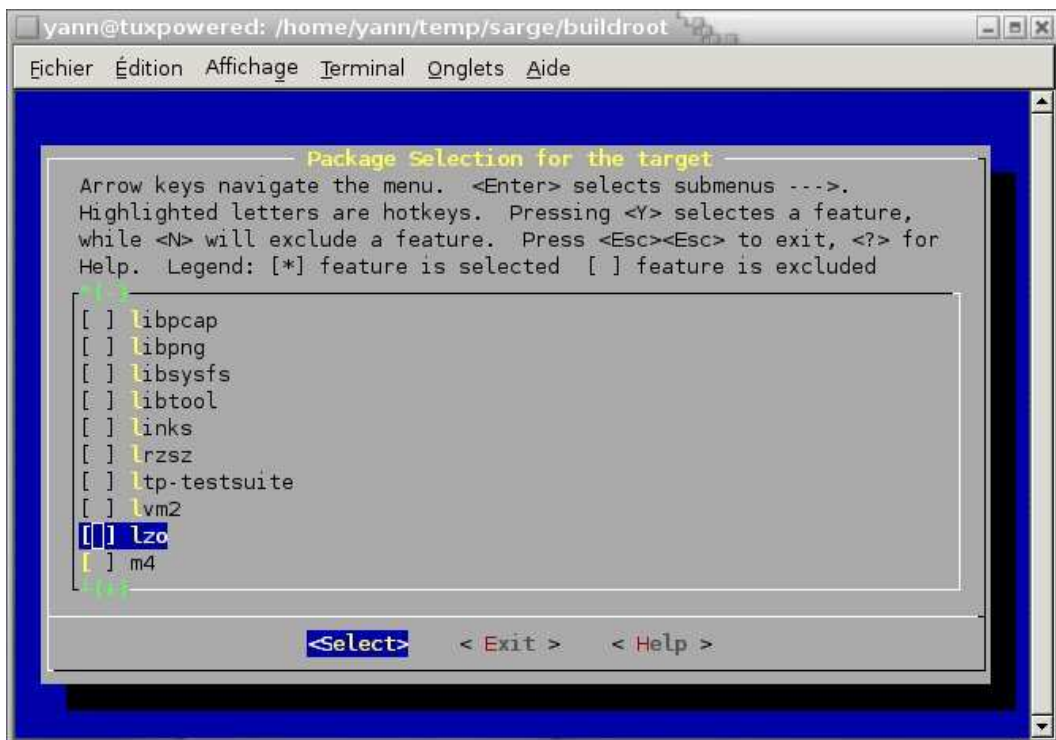
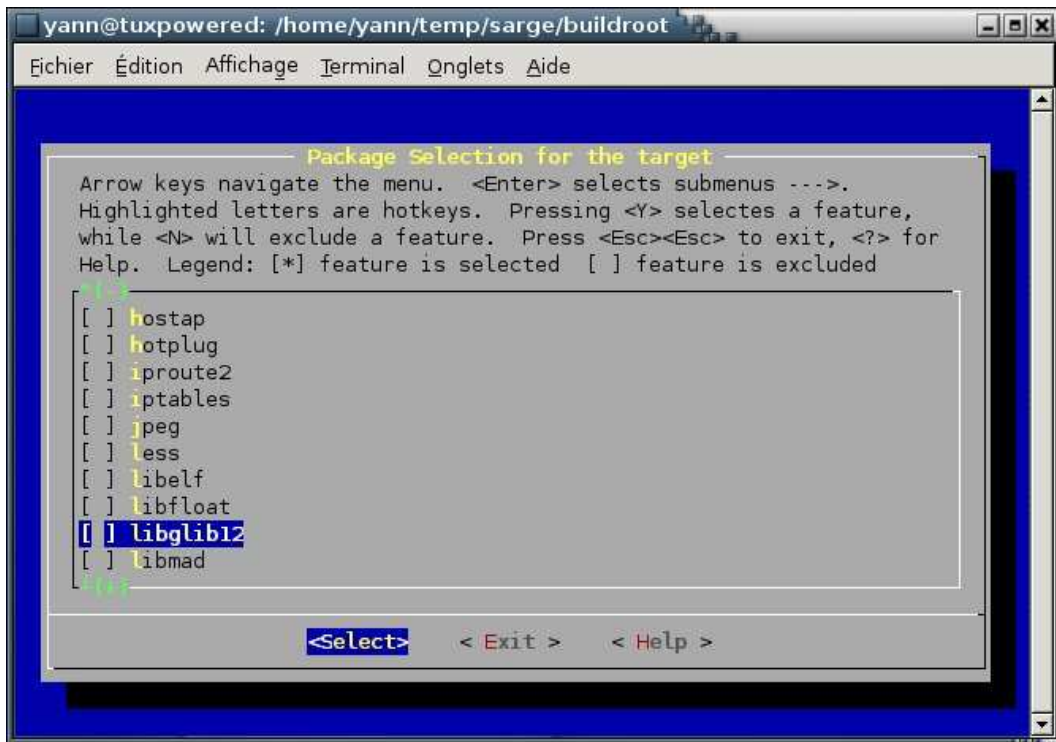
<Select>  < Exit >  < Help >
```



```
yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Package Selection for the target --
Arrow keys navigate the menu.  <Enter> selects submenus --->.
Highlighted letters are hotkeys.  Pressing <Y> selects a feature,
while <N> will exclude a feature.  Press <Esc><Esc> to exit, <?> for
Help.  Legend: [*] feature is selected [ ] feature is excluded
^~^~
[ ] distcc
[ ] dm
[ ] dnsmasq
[ ] dropbear_sshd
[ ] e2fsprogs
[ ] fakeroot
[ ] file
[ ] freetype
[*] gettext
[ ] gzip
^~^~

<Select>  < Exit >  < Help >
```



```
yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

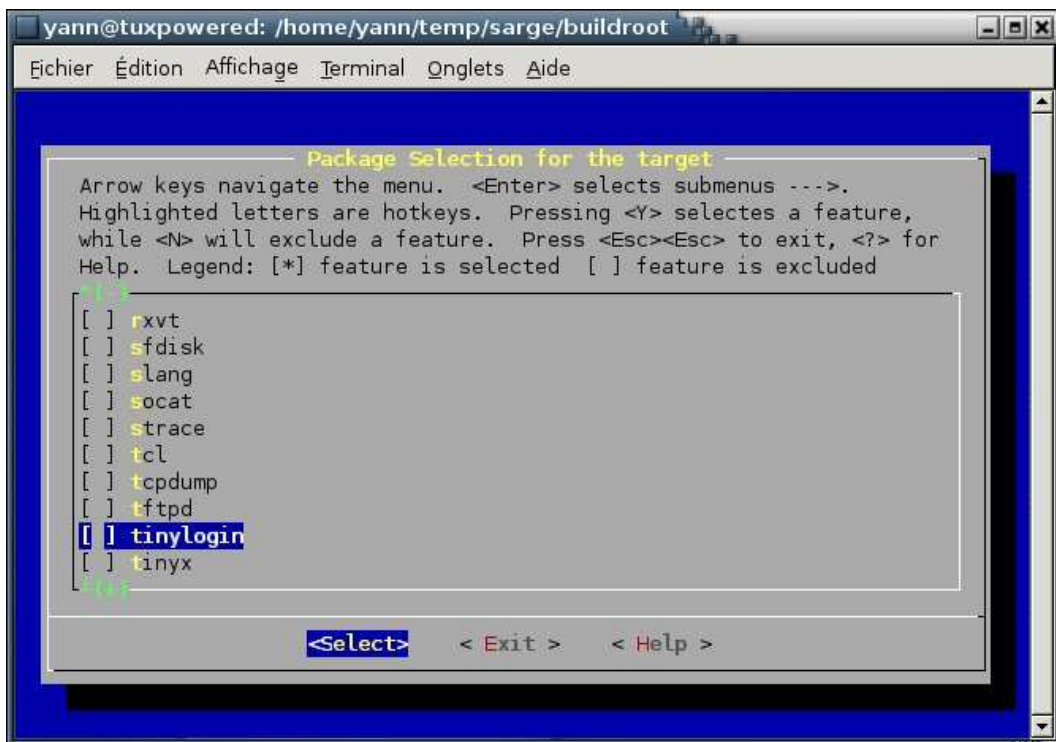
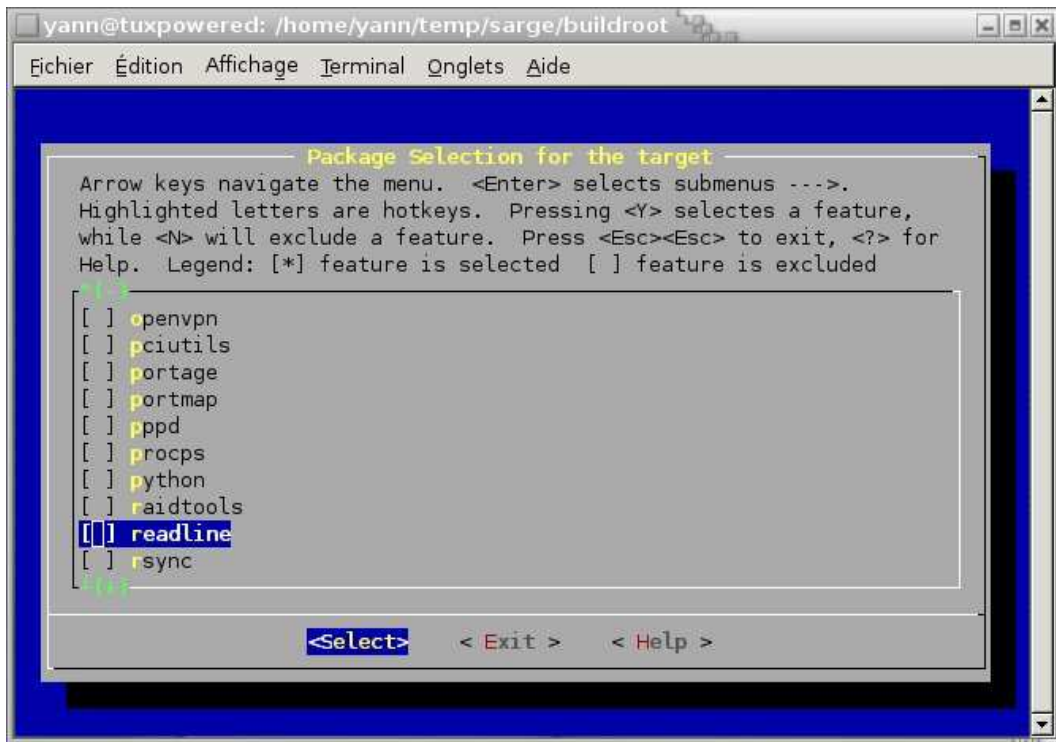
-- Package Selection for the target --
Arrow keys navigate the menu. <Enter> selects submenus ---.
Highlighted letters are hotkeys. Pressing <Y> selects a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded
^~^
[ ] microcom
[ ] microperl
[ ] microwin
[ ] mkdosfs
[ ] module-init-tools
[ ] modutils
[ ] mpg123
[ ] mrouted
[ ] mtd/jffs2 utilities
[*] nano
^~^

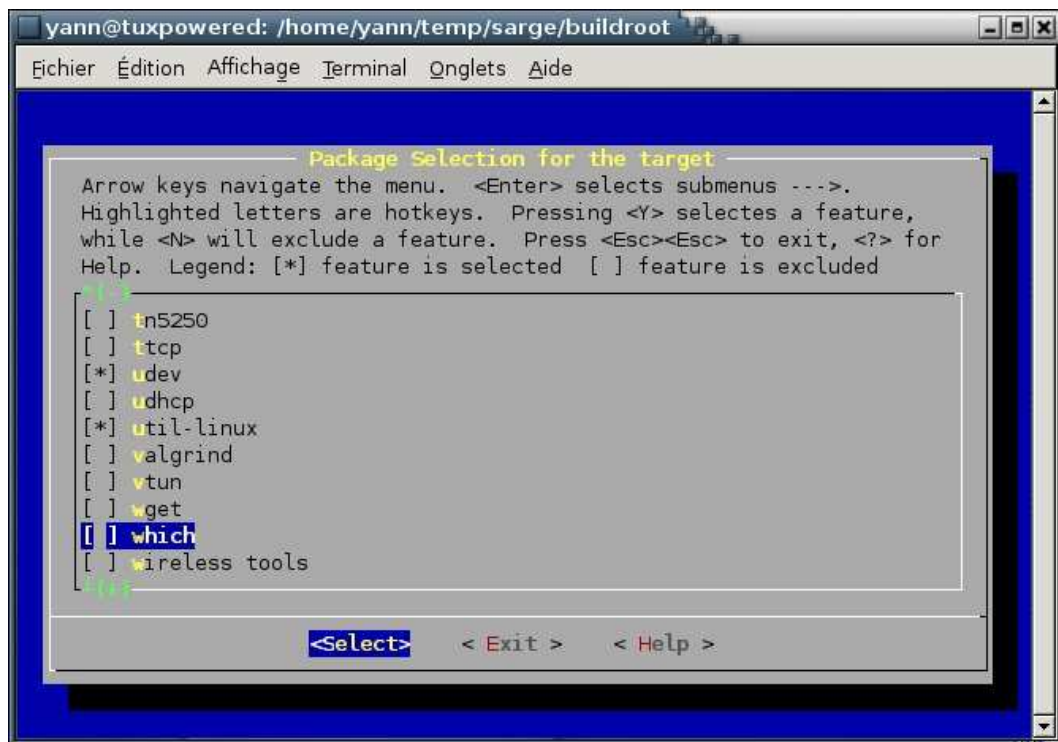
<Select>  < Exit >  < Help >
```

```
yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Package Selection for the target --
Arrow keys navigate the menu. <Enter> selects submenus ---.
Highlighted letters are hotkeys. Pressing <Y> selects a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded
^~^
[ ] ncurses
[ ] netkitbase
[ ] netkittelnet
[ ] netsnmp
[ ] newt
[ ] ntp
[ ] openNTPD
[*] openssh
-[-] openssl
[ ] openssl headers in target
^~^

<Select>  < Exit >  < Help >
```

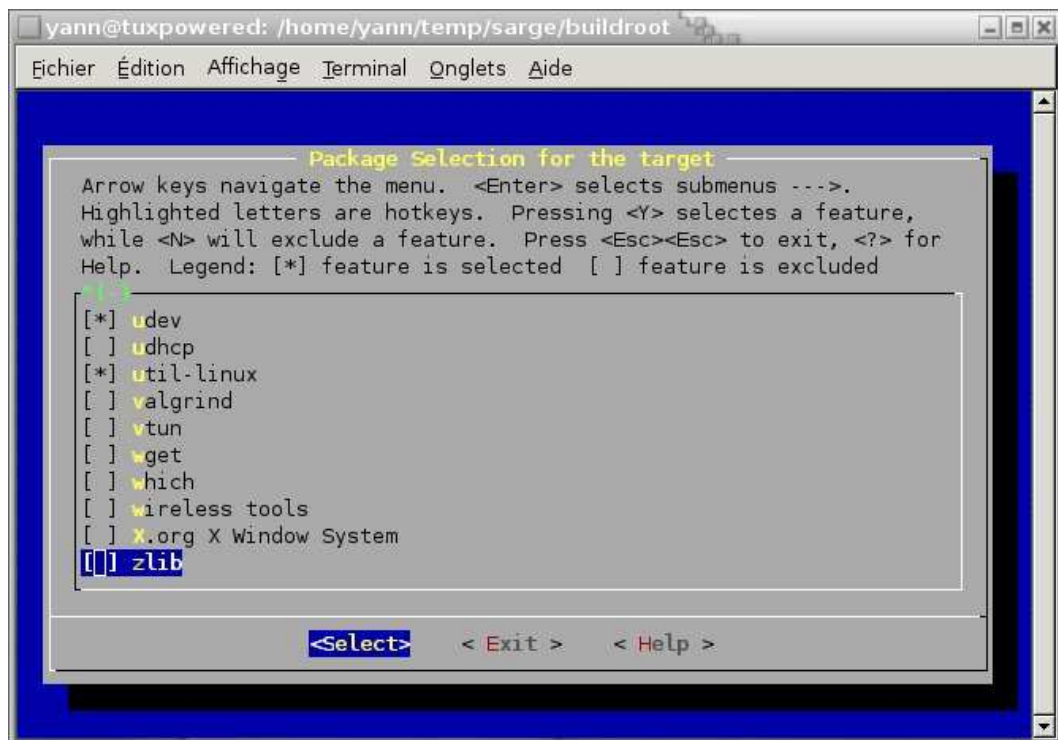




```
yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Package Selection for the target --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded
-+-+
[ ] tn5250
[ ] tcp
[*] udev
[ ] udhcp
[*] util-linux
[ ] valgrind
[ ] vtun
[ ] wget
[ ] which
[ ] wireless tools
-+-+

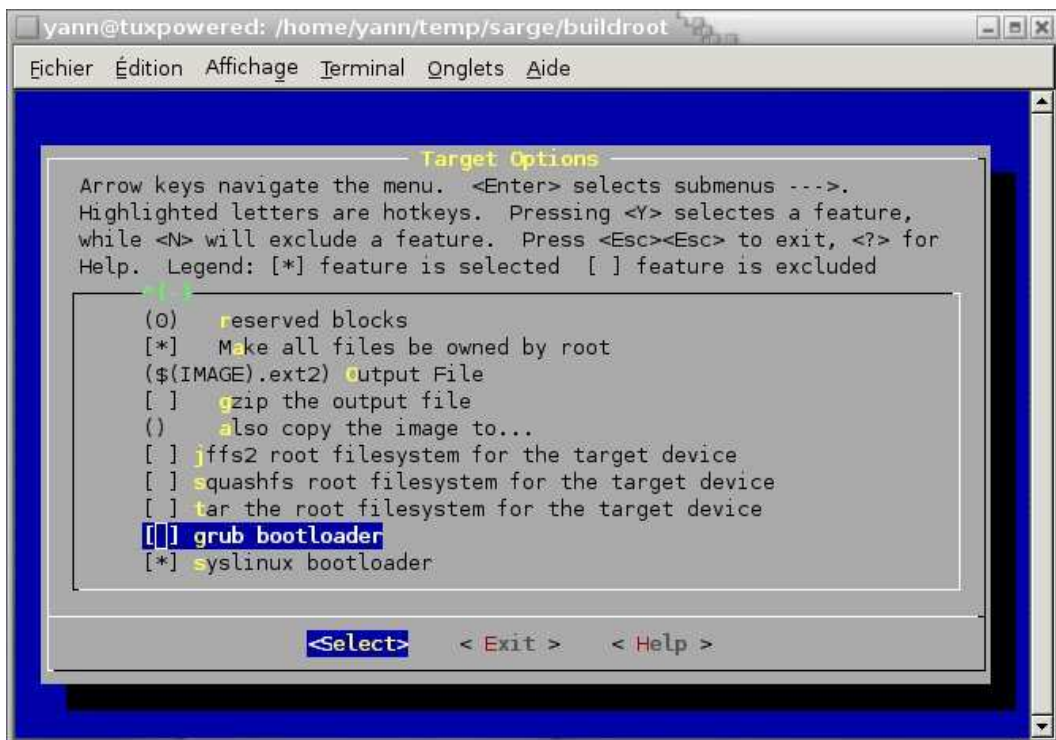
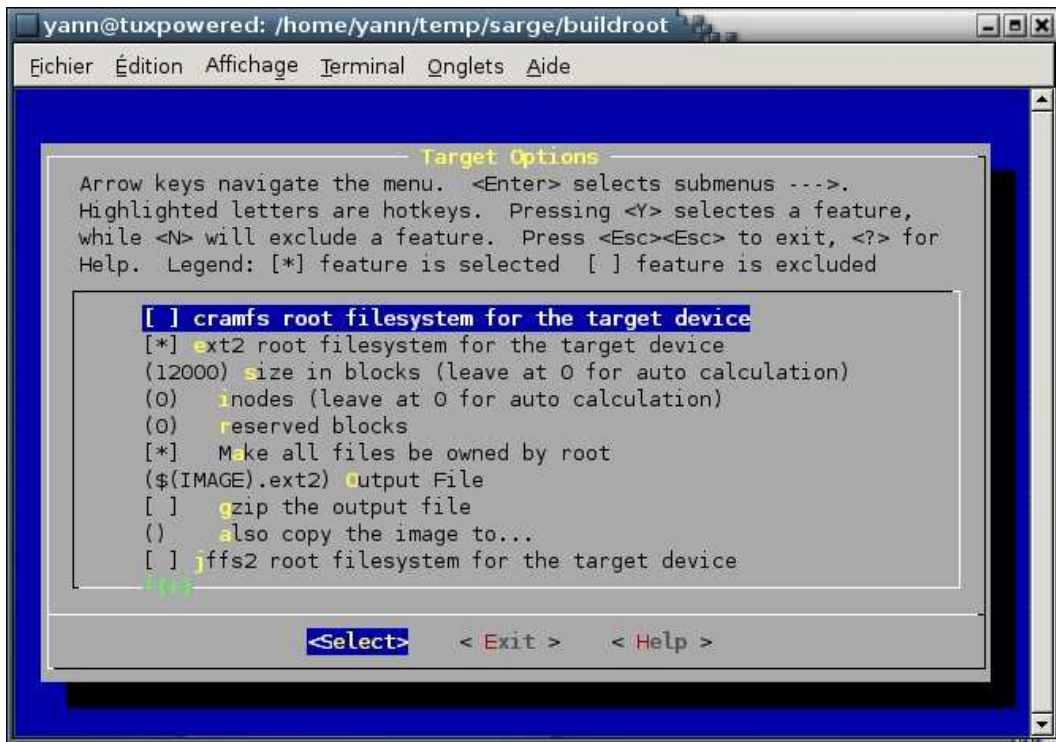
<Select>  < Exit >  < Help >
```

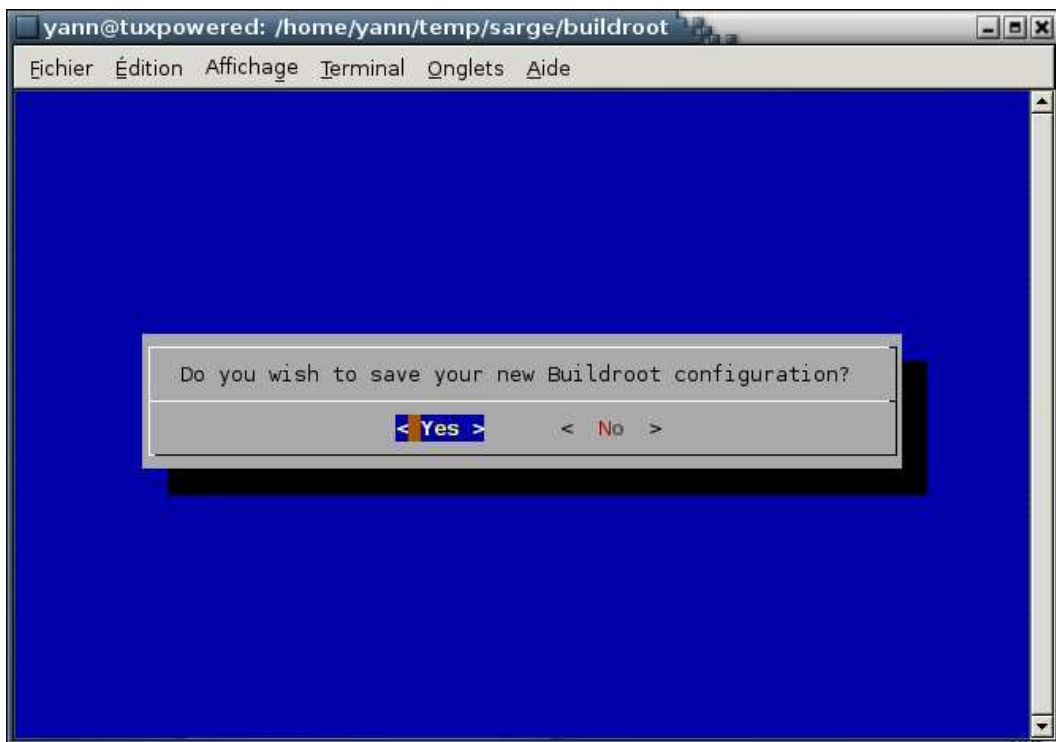
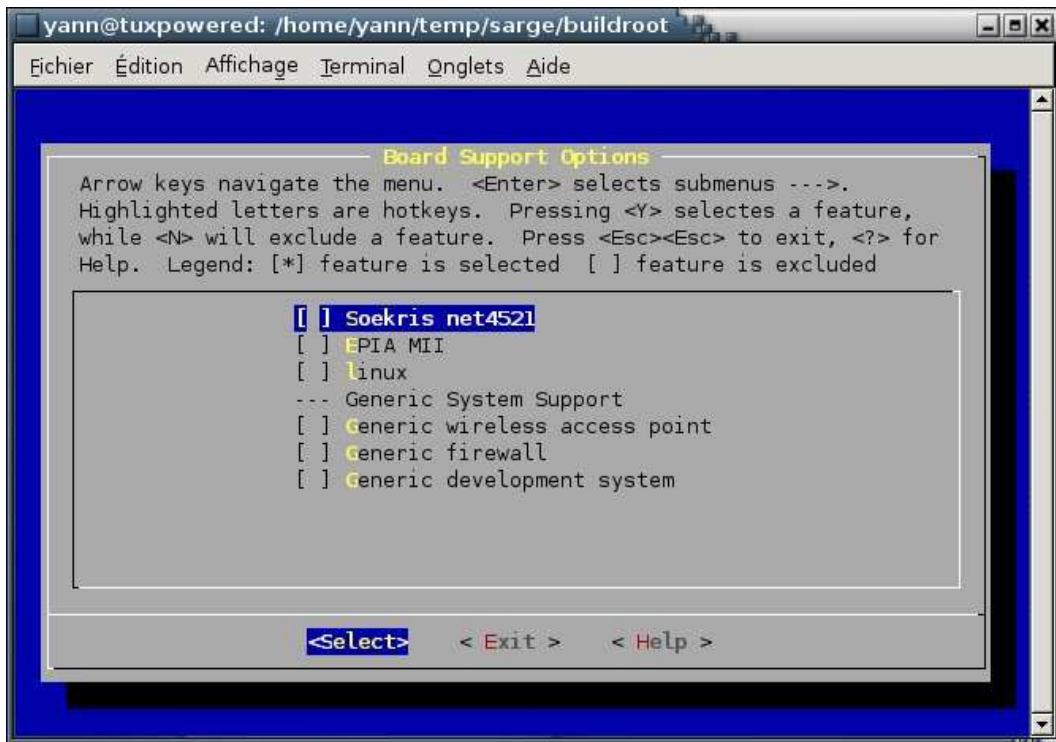


```
yann@tuxpowered: /home/yann/temp/sarge/buildroot
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Package Selection for the target --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help. Legend: [*] feature is selected [ ] feature is excluded
-+-+
[*] udev
[ ] udhcp
[*] util-linux
[ ] valgrind
[ ] vtun
[ ] wget
[ ] which
[ ] wireless tools
[ ] X.org X Window System
[ ] zlib
-+-+

<Select>  < Exit >  < Help >
```





Vous avez pu remarquer que nous allons construire la busybox mais que l'outil de configuration ne nous a pas laissé la configurer. Tout ceci est normal. D'après la documentation de buildroot <http://buildroot.uclibc.org/buildroot.html> :

1. on doit faire une première compilation sans essayer de configurer busybox.
2. on vane le répertoire build_ARCH/busybox/ et on lance make menuconfig. On peut alors configurer busybox a volonté.
3. On copie le fichier .config dans package/busybox/busybox.config afin que la configuration soit prise en compte.

4. On relance la compilation.

On va donc lancer notre première compilation :

```
yann@tuxpowered:~/temp/sarge/buildroot$ make
```

```
....
```

et quelques cafés plus tard....

```
....
```

```
/home/yann/temp/sarge/buildroot/build_i386/genext2fs-1.3/genext2fs \
```

```
-d /home/yann/temp/sarge/buildroot/build_i386/root \
```

```
-D target/generic/device_table.txt \
```

```
-U -b 12000 \
```

```
/home/yann/temp/sarge/buildroot/root_fs_i386.ext2
```

```
number of inodes too low, increasing to 545
```

```
-rw-r--r-- 1 yann yann 12288000 2005-07-14 15:42 /home/yann/temp/sarge/buildroot/root_f
```

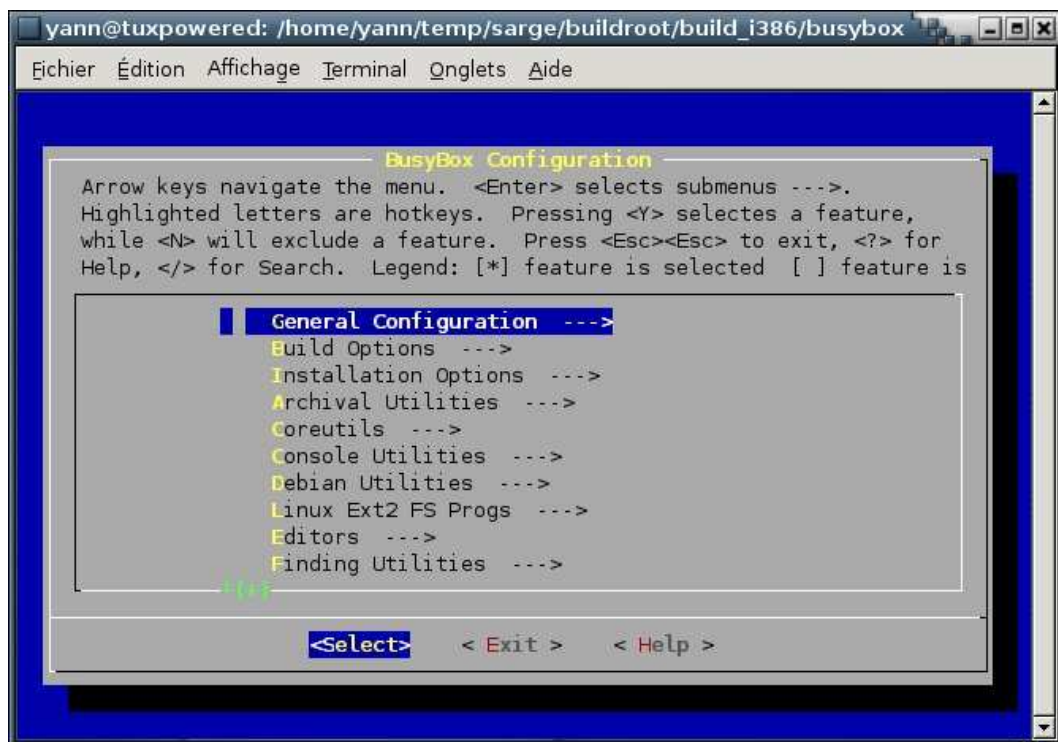
```
yann@tuxpowered:~/temp/sarge/buildroot$
```

Notre première image du système de fichier est créée. On va maintenant configurer busybox :

```
yann@tuxpowered:~/temp/sarge/buildroot$ cd build_i386/busybox/
```

```
yann@tuxpowered:~/temp/sarge/buildroot/build_i386/busybox$ make menuconfig
```

Voici mon fichier de configuration [fichier_utils/last/config-busybox](#) Voici la configuration en image :



```

yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- General Configuration --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is

[ ] Buffer allocation policy (Allocate on the Stack) --->
[*] Show verbose applet usage messages
[ ] Support --install [-s] to install applet links at runtime
[*] Enable locale support (system needs locale for this to work)
[ ] Support for devfs
[*] Use the devpts filesystem for Unix98 PTYS
[ ] Clean up all memory before exiting (usually not needed)
--- Support for SUID/SGID handling
[ ] Runtime SUID/SGID configuration via /etc/busybox.conf
[ ] Support NSA Security Enhanced Linux

<Select>  < Exit >  < Help >

```

```

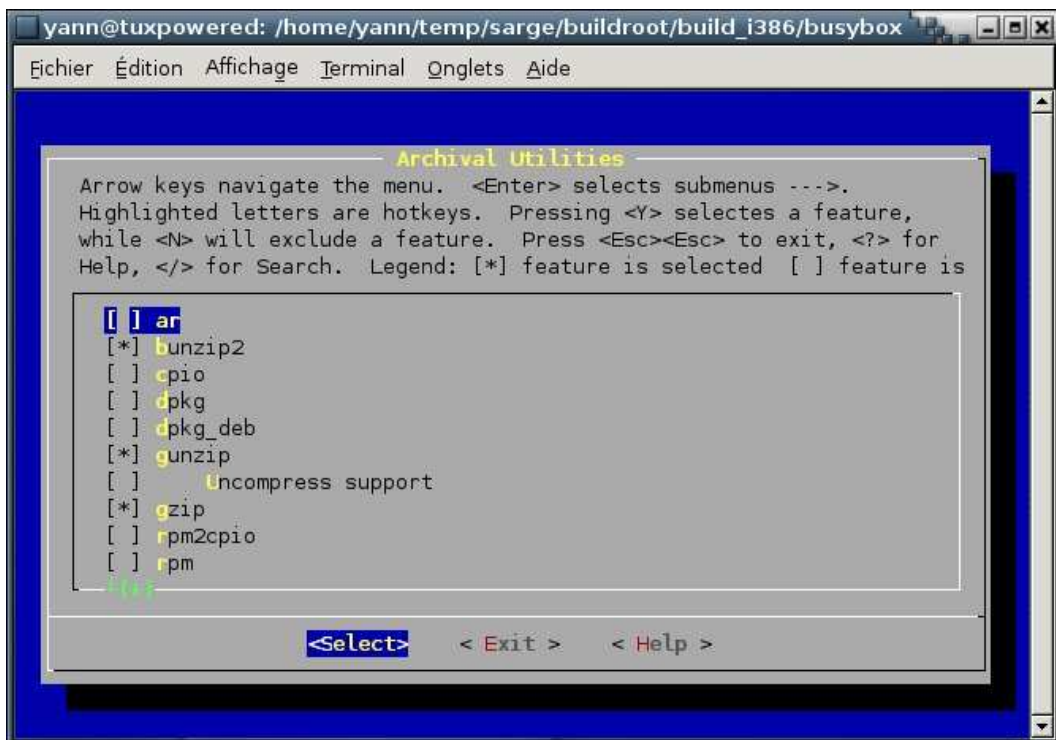
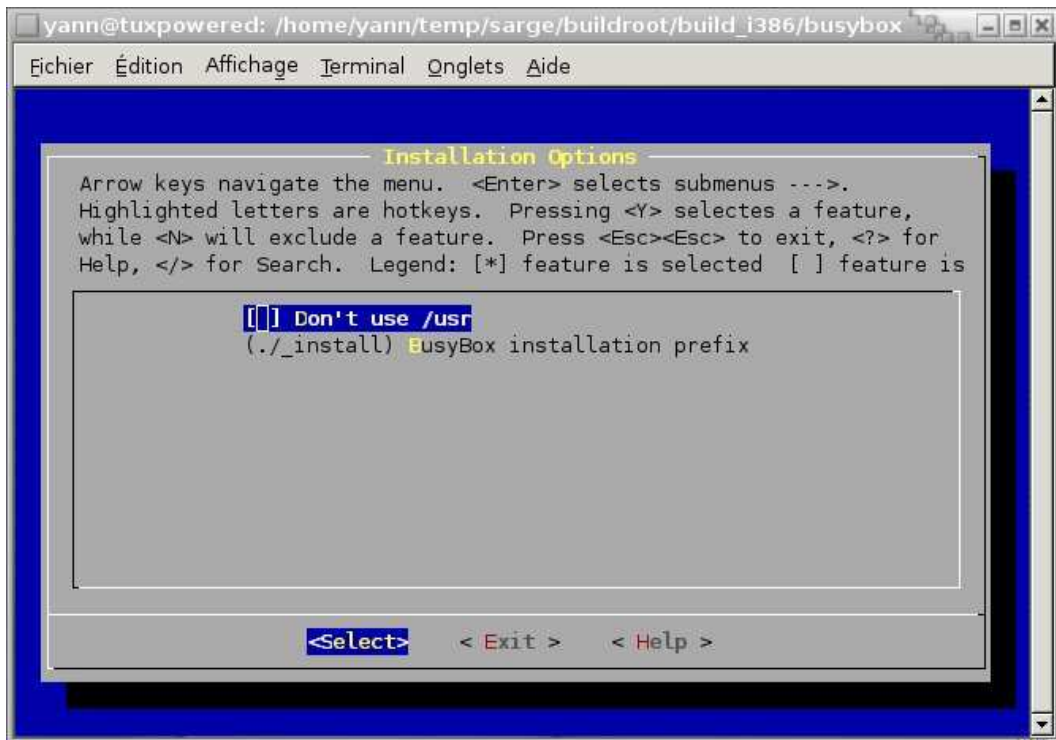
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

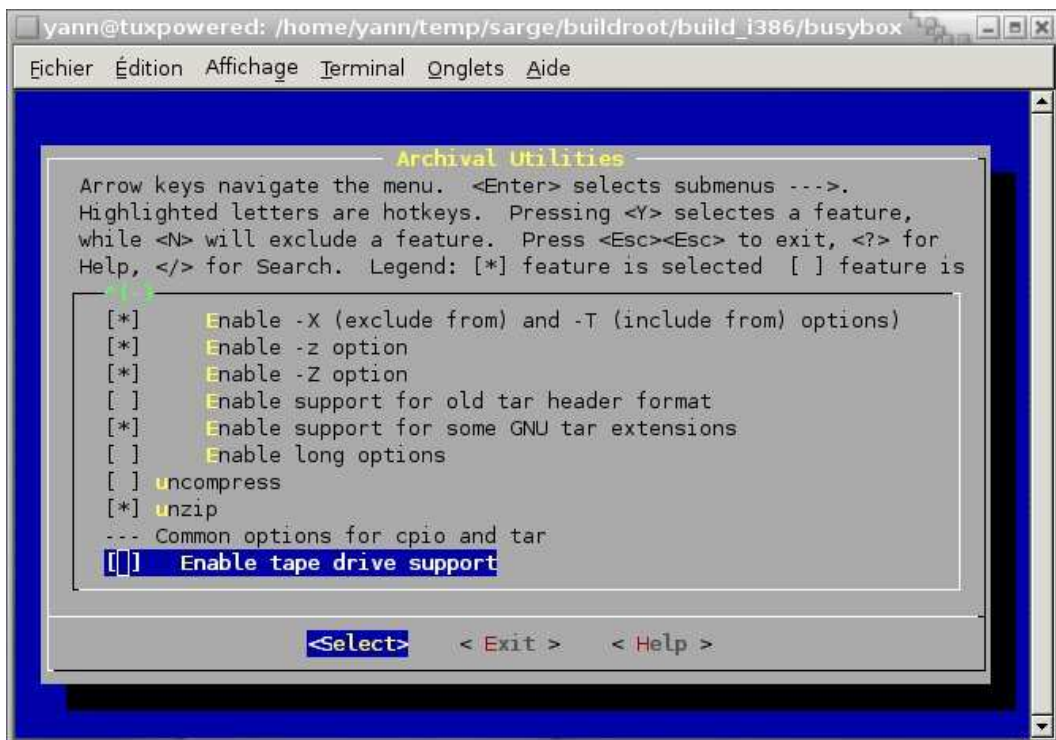
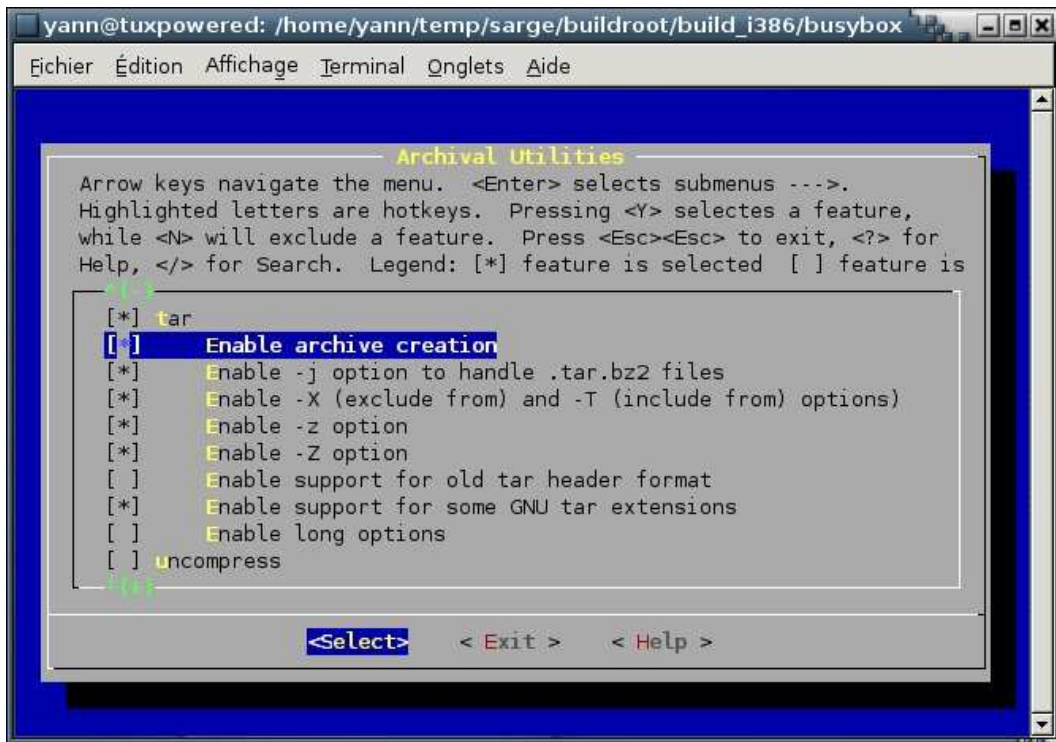
-- Build Options --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is

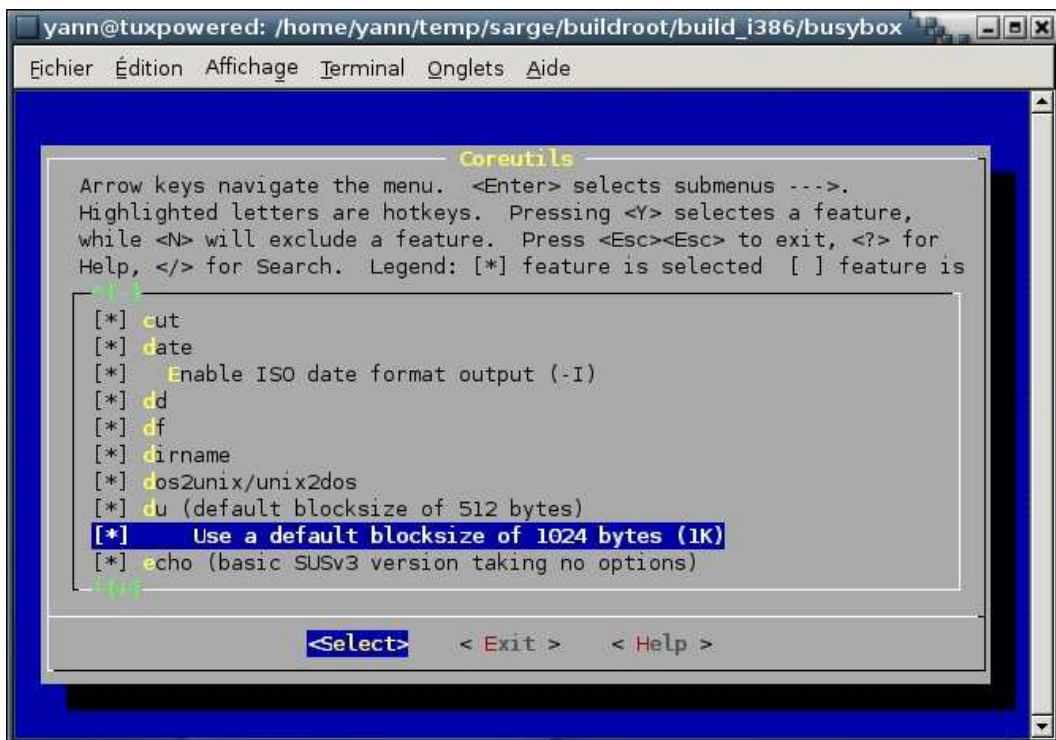
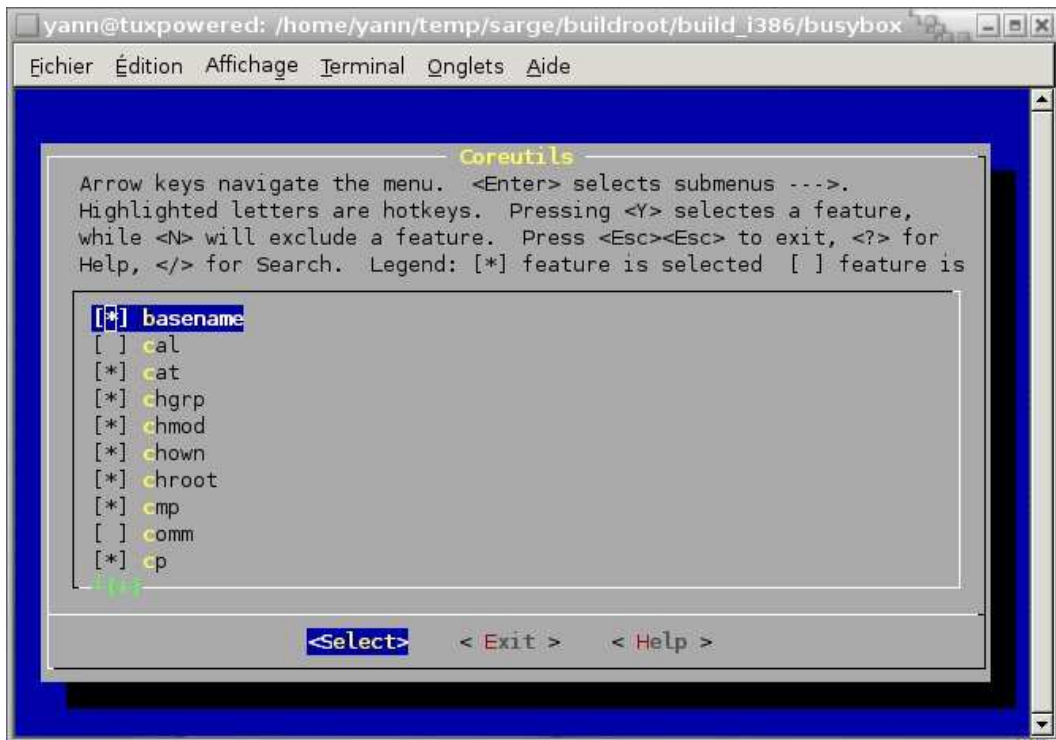
[ ] Build BusyBox as a static binary (no shared libs)
[*] Build with Large File Support (for accessing files > 2 GB)
[ ] Do you want to build BusyBox with a Cross Compiler?
() Any extra CFLAGS options for the compiler?

<Select>  < Exit >  < Help >

```







```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Coreutils
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not

[*] echo (basic SUSv3 version taking no options)
[*]   Enable echo options (-n and -e)
[*] env
[ ] printenv
[*] expr
[*] false
[ ] fold
[*] head
[[*]] Enable head options (-c, -q, and -v)
[*] hostid

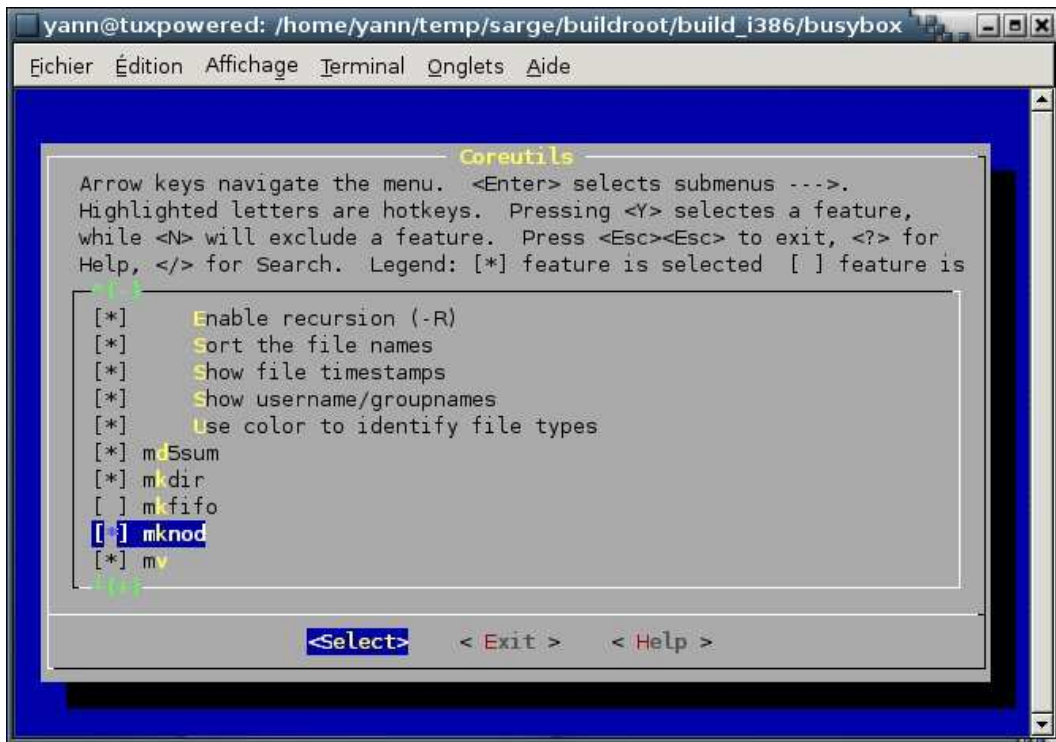
+-----+
|<Select>| < Exit > | < Help > |
+-----+
```

```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Coreutils
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not

[ ]   Enable head options (-c, -q, and -v)
[*] hostid
[*] id
[*] install
[ ] length
[*] ln
[*] logname
[*] ls
[[*]] Enable filetyping options (-p and -F)
[*]   Enable symlinks dereferencing (-L)

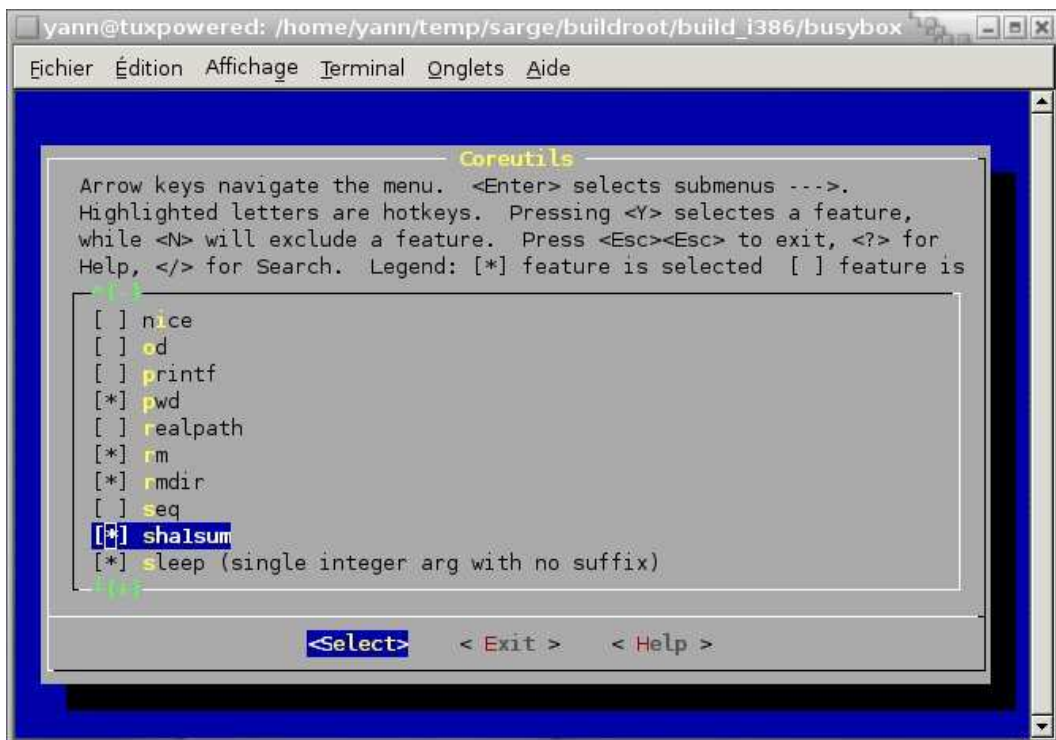
+-----+
|<Select>| < Exit > | < Help > |
+-----+
```

```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Coreutils
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not
[*] Enable recursion (-R)
[*] Sort the file names
[*] Show file timestamps
[*] Show username/groupnames
[*] Use color to identify file types
[*] md5sum
[*] mkdir
[ ] mkfifo
[*] mknod
[*] mv
+---+

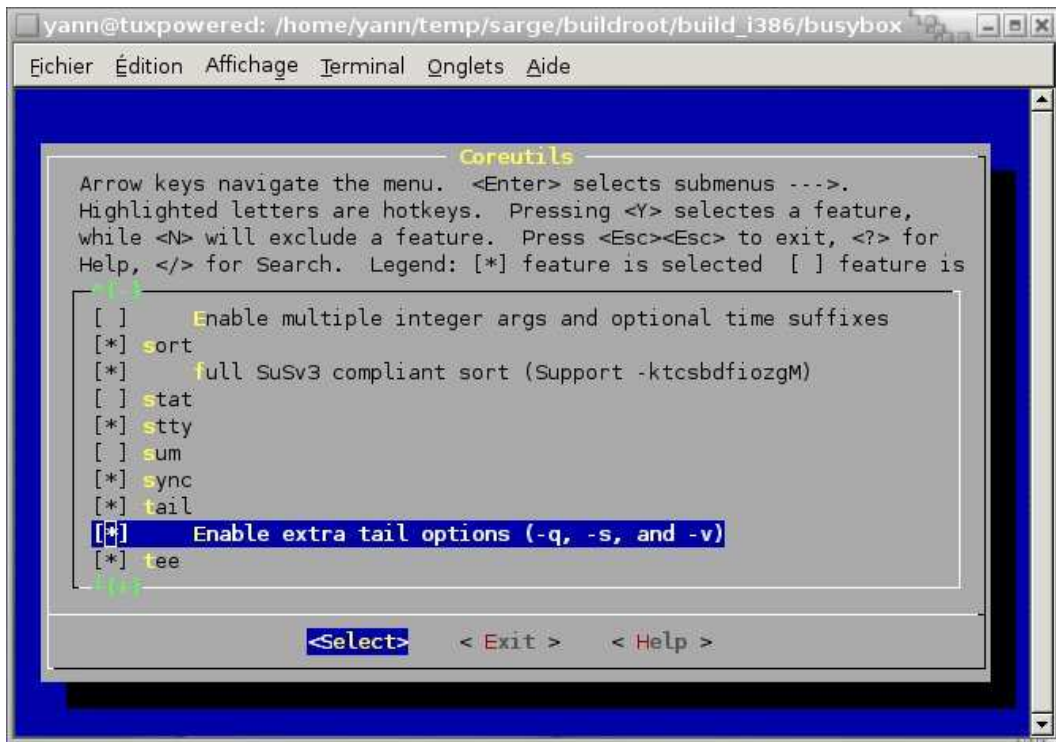
<Select>  < Exit >  < Help >
```



```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Coreutils
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not
[ ] nice
[ ] od
[ ] printf
[*] pwd
[ ] realpath
[*] rm
[*] rmdir
[ ] seq
[*] shasum
[*] sleep (single integer arg with no suffix)
+---+

<Select>  < Exit >  < Help >
```

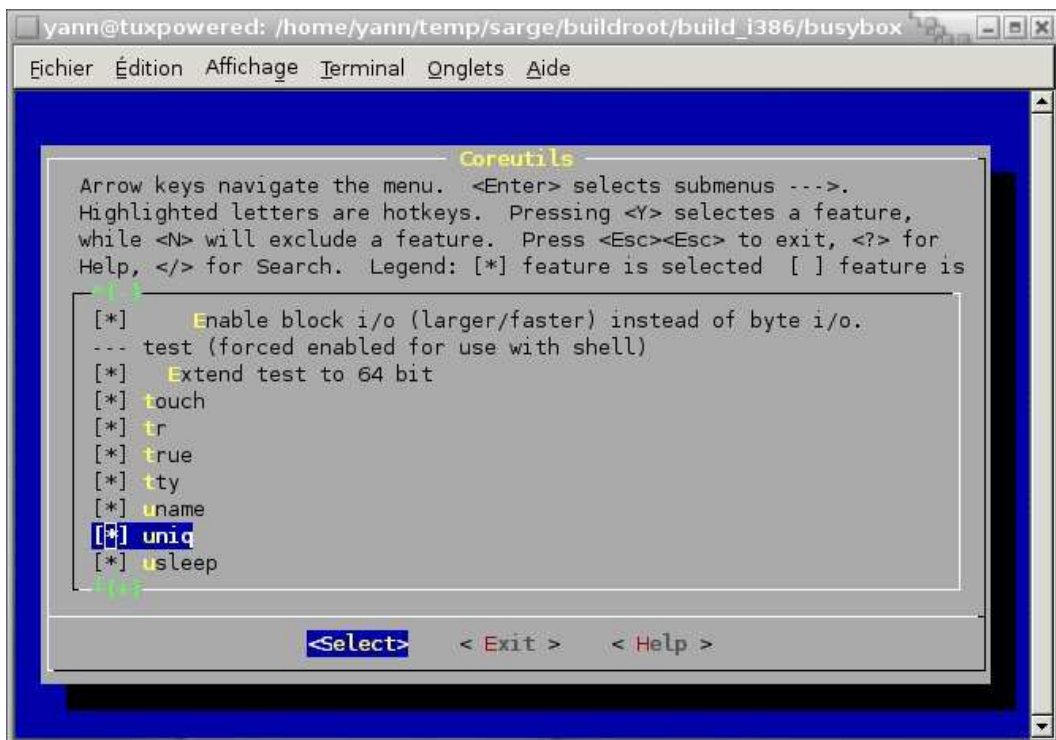


```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Coreutils
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not

[ ] Enable multiple integer args and optional time suffixes
[*] sort
[*] full SuSv3 compliant sort (Support -ktsbdfiozgM)
[ ] stat
[*] stty
[ ] sum
[*] sync
[*] tail
[*] Enable extra tail options (-q, -s, and -v)
[*] tee

+-----+
|<Select>| < Exit > | < Help > |
+-----+
```

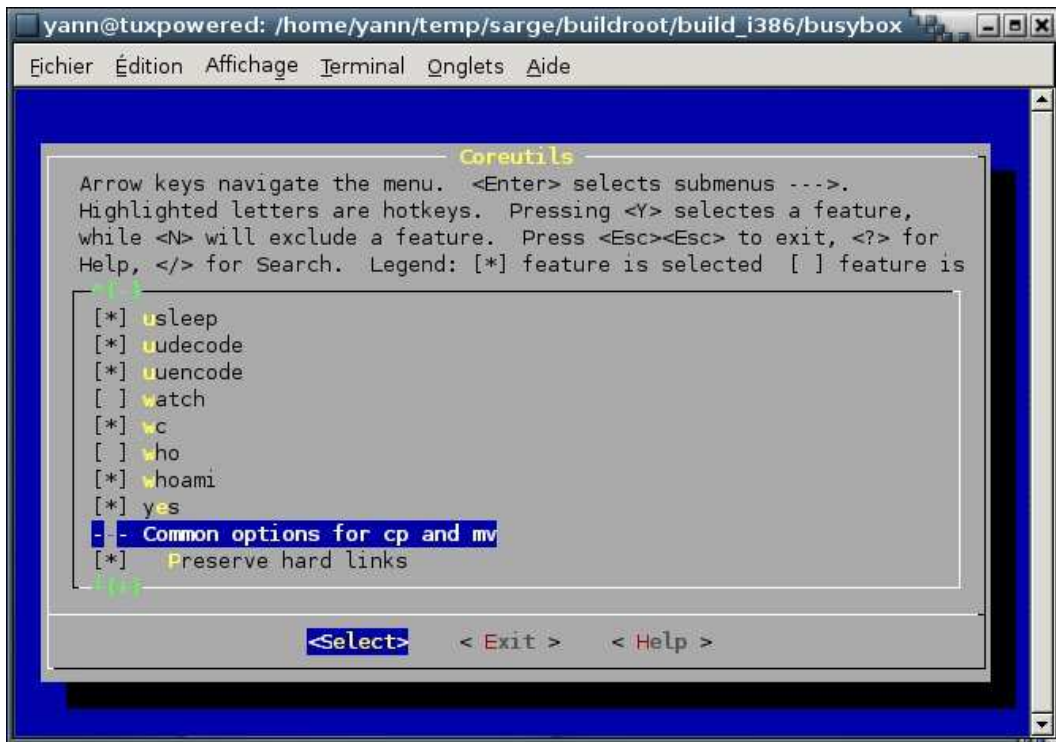


```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Coreutils
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not

[*] Enable block i/o (larger/faster) instead of byte i/o.
--- test (forced enabled for use with shell)
[*] Extend test to 64 bit
[*] touch
[*] tr
[*] true
[*] tty
[*] uname
[*] uniq
[*] usleep

+-----+
|<Select>| < Exit > | < Help > |
+-----+
```

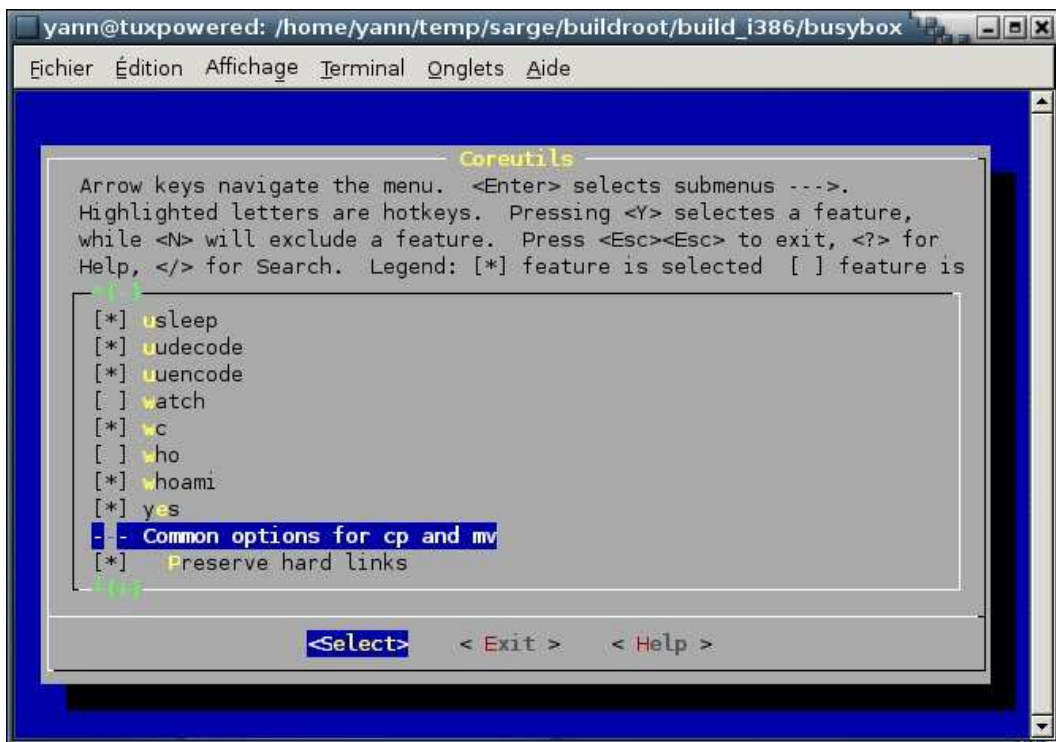


```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Coreutils
Arrow keys navigate the menu.  <Enter> selects submenus --->.
Highlighted letters are hotkeys.  Pressing <Y> selectes a feature,
while <N> will exclude a feature.  Press <Esc><Esc> to exit, <?> for
Help, </> for Search.  Legend: [*] feature is selected [ ] feature is
not

[*]  usleep
[*]  uudecode
[*]  uuencode
[ ]  watch
[*]  wc
[ ]  who
[*]  whoami
[*]  yes
- - Common options for cp and mv
[*]  Preserve hard links

<Select>  < Exit >  < Help >
```

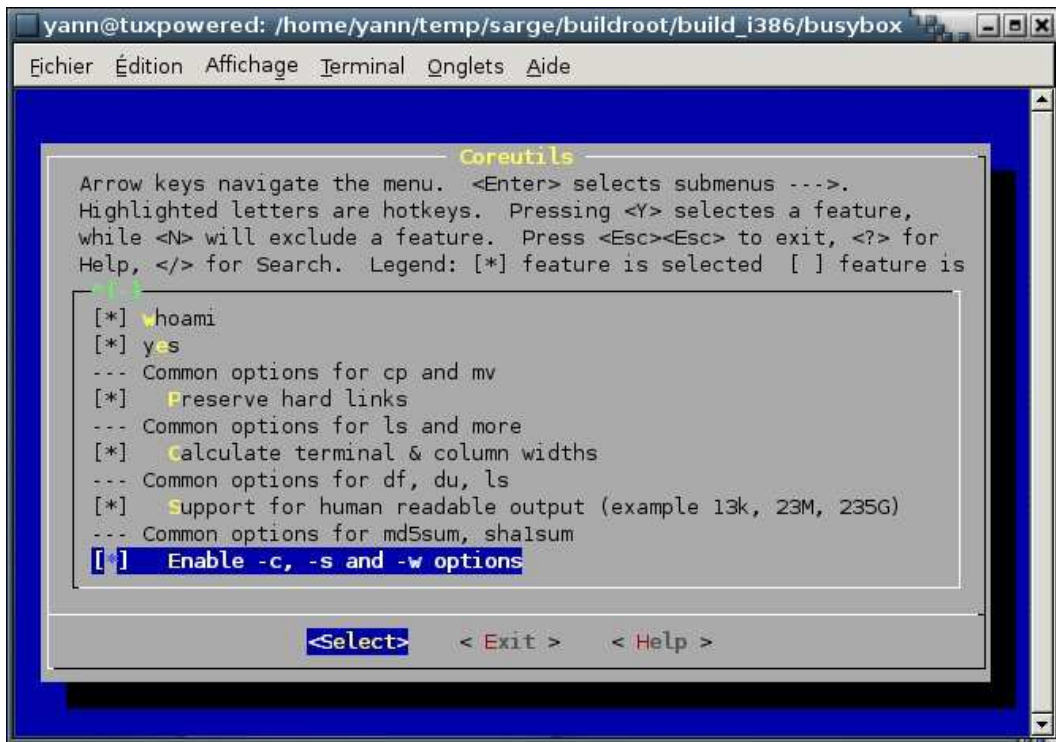


```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Coreutils
Arrow keys navigate the menu.  <Enter> selects submenus --->.
Highlighted letters are hotkeys.  Pressing <Y> selectes a feature,
while <N> will exclude a feature.  Press <Esc><Esc> to exit, <?> for
Help, </> for Search.  Legend: [*] feature is selected [ ] feature is
not

[*]  usleep
[*]  uudecode
[*]  uuencode
[ ]  watch
[*]  wc
[ ]  who
[*]  whoami
[*]  yes
- - Common options for cp and mv
[*]  Preserve hard links

<Select>  < Exit >  < Help >
```

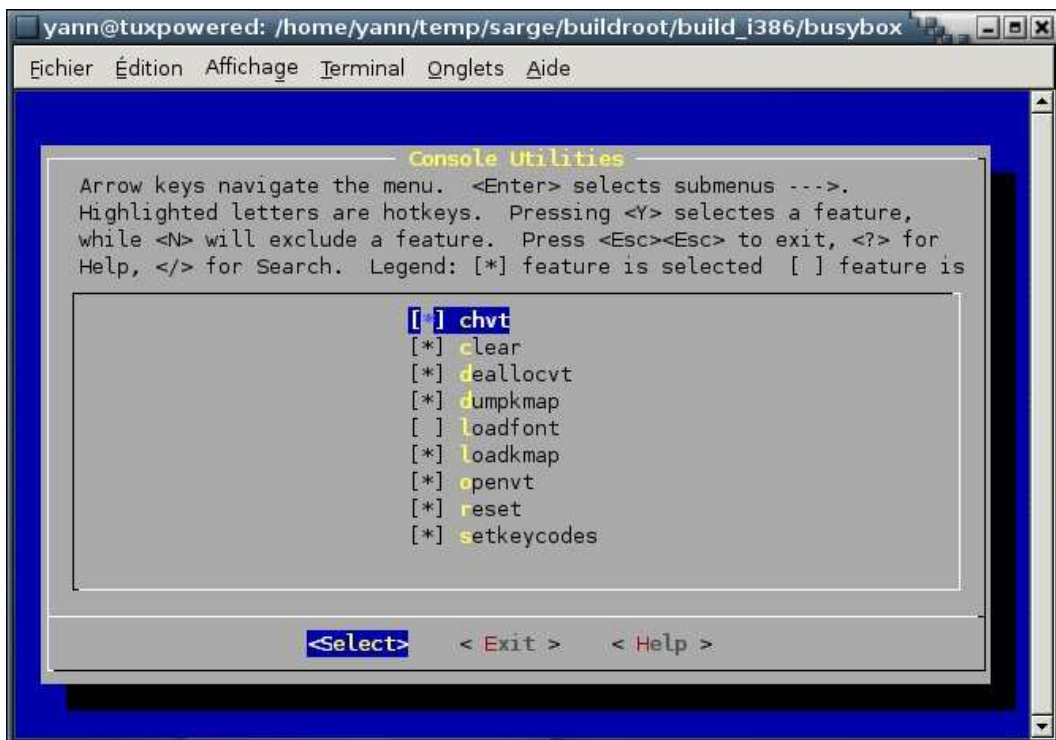


```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Coreutils --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not

[*] whoami
[*] yes
--- Common options for cp and mv
[*] Preserve hard links
--- Common options for ls and more
[*] calculate terminal & column widths
--- Common options for df, du, ls
[*] Support for human readable output (example 13k, 23M, 235G)
--- Common options for md5sum, shasum
[*] Enable -c, -s and -w options

<Select>  < Exit >  < Help >
```

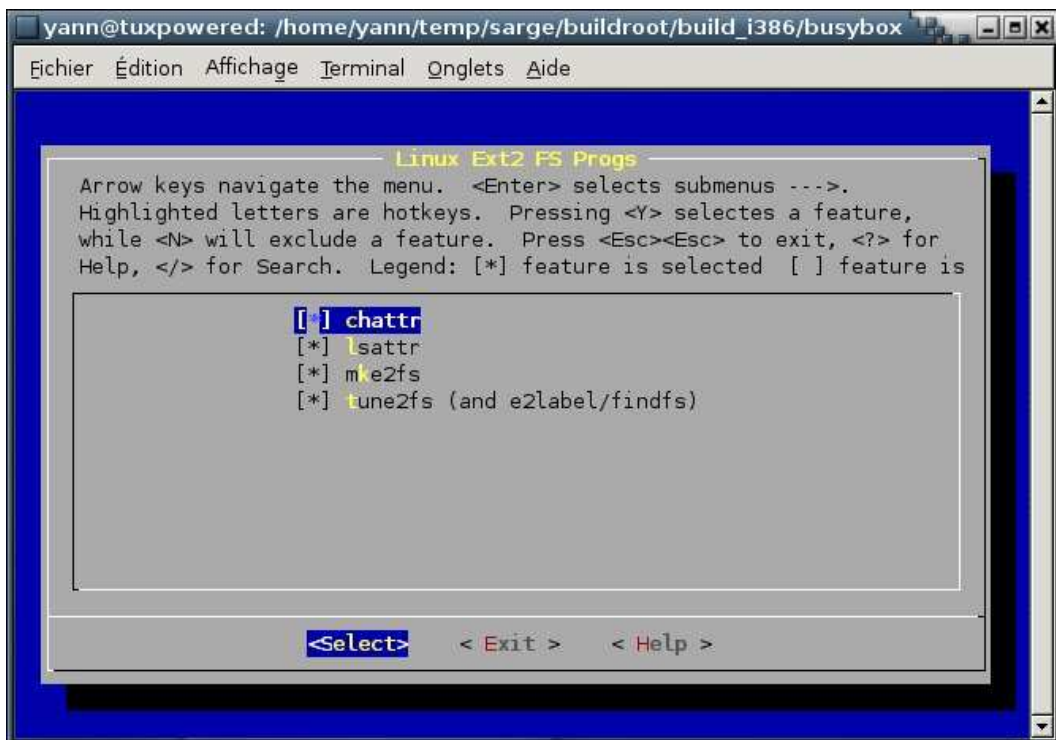
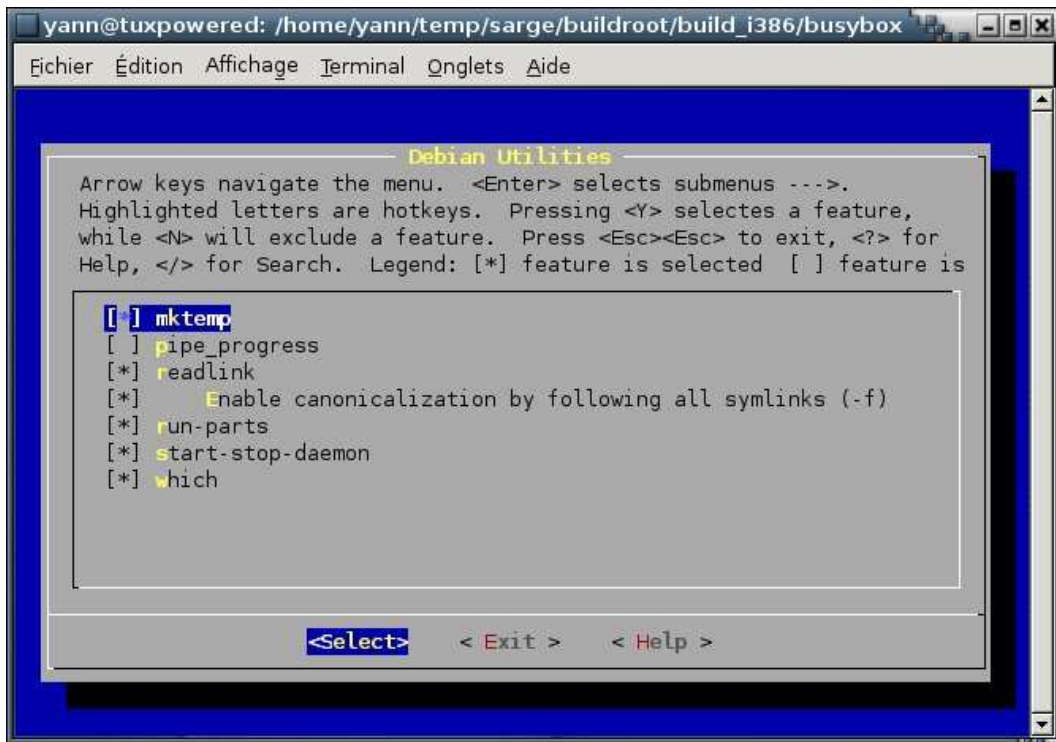


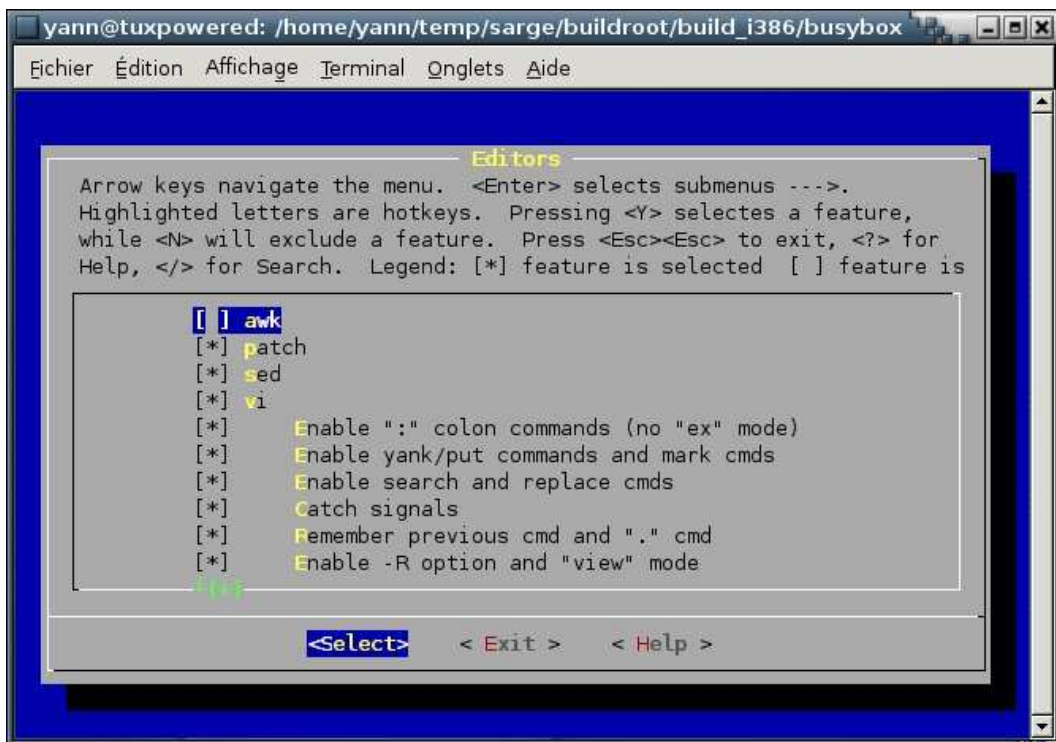
```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Console Utilities --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not

[*] chvt
[*] clear
[*] deallocvt
[*] dumpkmap
[ ] loadfont
[*] loadkmap
[*] openvt
[*] reset
[*] setkeycodes

<Select>  < Exit >  < Help >
```



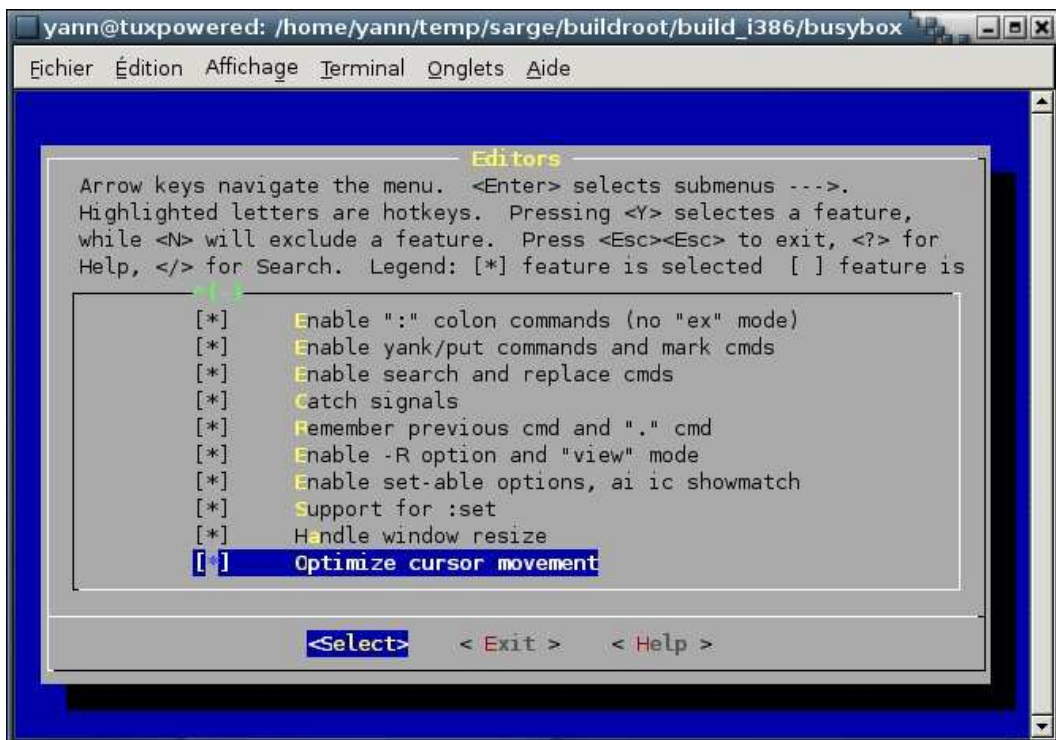


```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Editors
Arrow keys navigate the menu.  <Enter> selects submenus --->.
Highlighted letters are hotkeys.  Pressing <Y> selectes a feature,
while <N> will exclude a feature.  Press <Esc><Esc> to exit, <?> for
Help, </> for Search.  Legend: [*] feature is selected [ ] feature is

[ ] awk
[*] patch
[*] sed
[*] vi
[*] Enable ":" colon commands (no "ex" mode)
[*] Enable yank/put commands and mark cmds
[*] Enable search and replace cmds
[*] Catch signals
[*] Remember previous cmd and "." cmd
[*] Enable -R option and "view" mode

<Select>  < Exit >  < Help >
```



```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Editors
Arrow keys navigate the menu.  <Enter> selects submenus --->.
Highlighted letters are hotkeys.  Pressing <Y> selectes a feature,
while <N> will exclude a feature.  Press <Esc><Esc> to exit, <?> for
Help, </> for Search.  Legend: [*] feature is selected [ ] feature is

[*] Enable ":" colon commands (no "ex" mode)
[*] Enable yank/put commands and mark cmds
[*] Enable search and replace cmds
[*] Catch signals
[*] Remember previous cmd and "." cmd
[*] Enable -R option and "view" mode
[*] Enable set-able options, ai ic showmatch
[*] Support for :set
[*] Handle window resize
[*] Optimize cursor movement

<Select>  < Exit >  < Help >
```

```

yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

          Finding Utilities
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is

[*] find
[*] Enable modified time matching (-mtime) option
[*] Enable permissions matching (-perm) option
[*] Enable filetype matching (-type) option
[*] Enable stay in filesystem (-xdev) option
[ ] Enable -newer option for comparing file mtimes
[ ] Enable inode number matching (-inum) option
[*] grep
[*] Support extended regular expressions (egrep & grep -E)
[*] Alias fgrep to grep -f

<Select>  < Exit >  < Help >

```

```

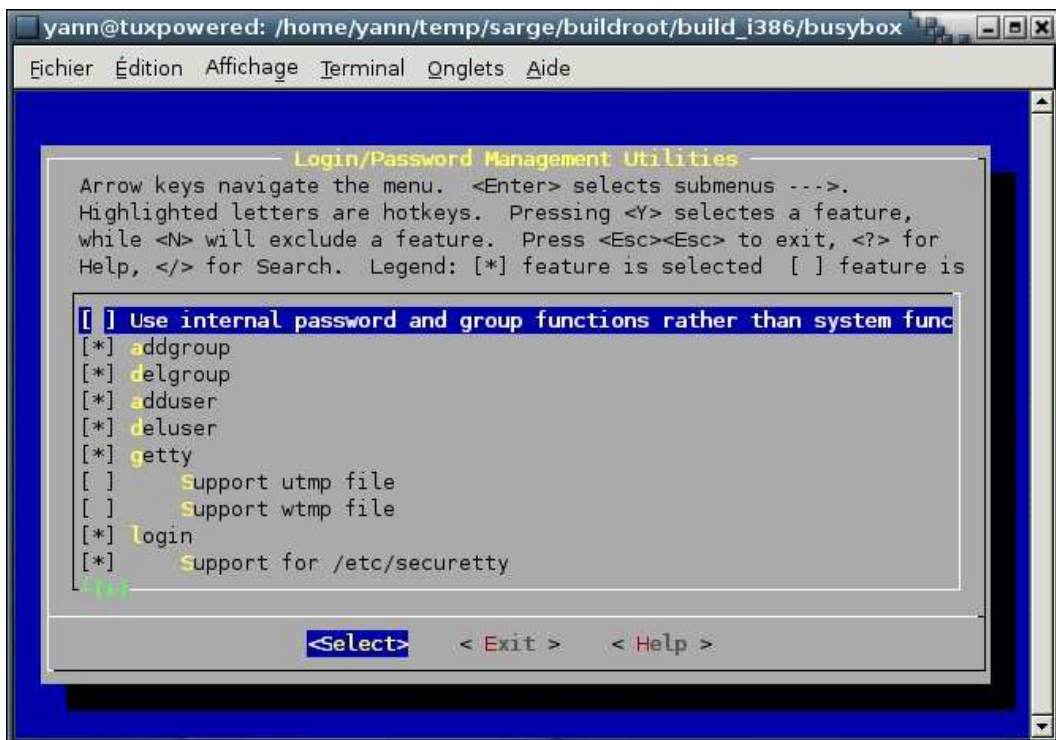
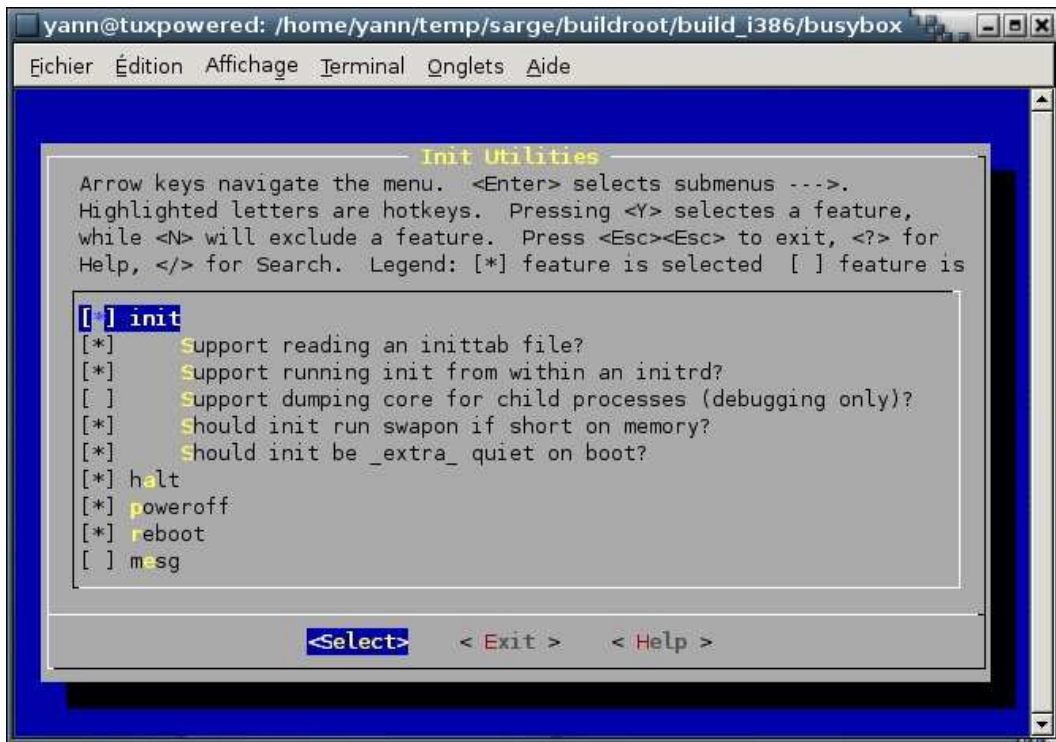
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

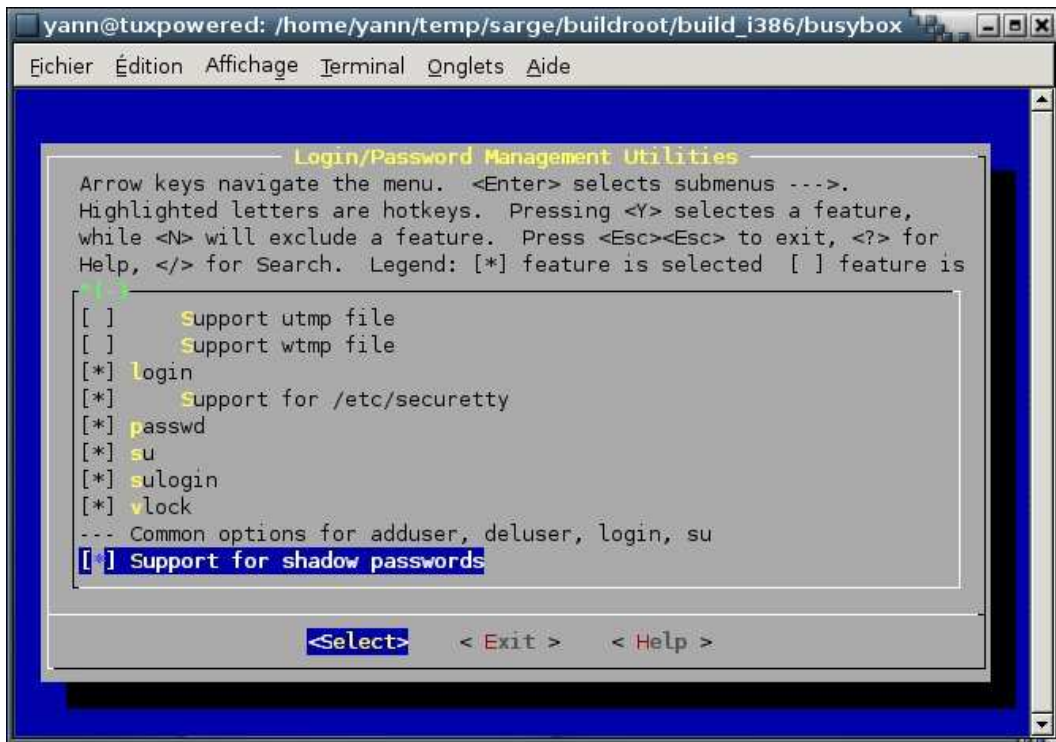
          Finding Utilities
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is

[ ] Enable inode number matching (-inum) option
[*] grep
[*] Support extended regular expressions (egrep & grep -E)
[*] Alias fgrep to grep -f
[*] Enable before and after context flags (-A, -B and -C)
[*] xargs
[ ] Enable prompt and confirmation option -p
[*] Enable support single and double quotes and backslash
[*] Enable support options -x
[*] Enable options -O

<Select>  < Exit >  < Help >

```

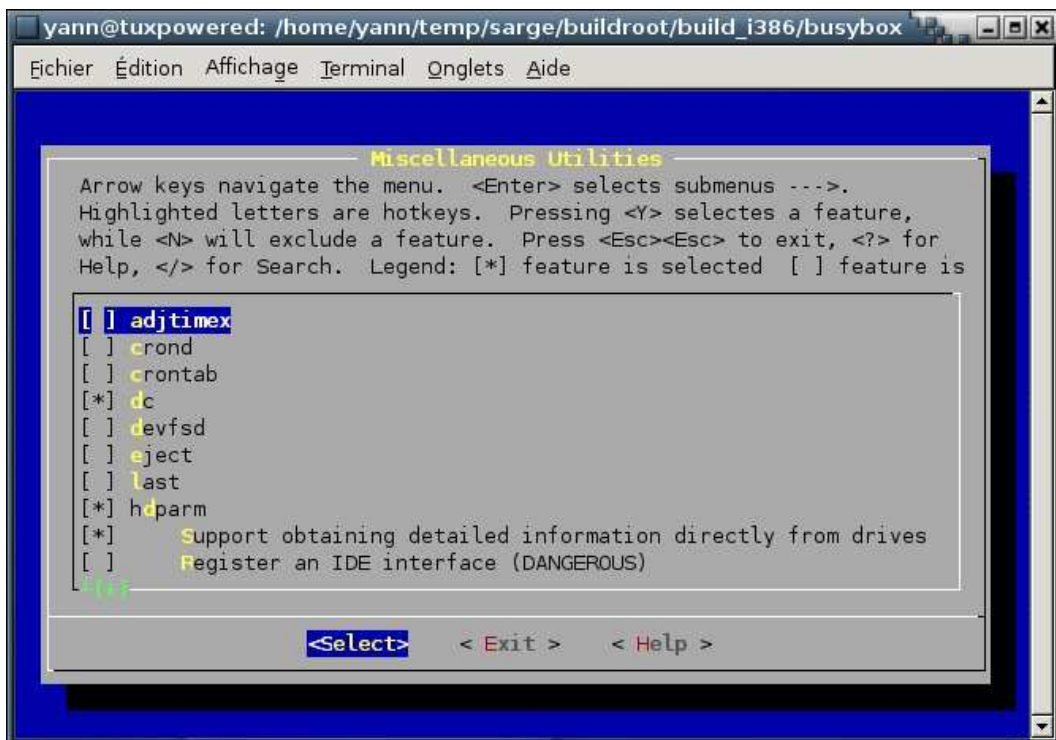





```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Login/Password Management Utilities --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not
[ ]   Support utmp file
[ ]   Support wtmp file
[*]  login
[*]   Support for /etc/security
[*]  passwd
[*]  su
[*]  slogin
[*]  vlock
--- Common options for adduser, deluser, login, su
[Y] Support for shadow passwords

<Select>  < Exit >  < Help >
```



```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Miscellaneous Utilities --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not
[Y] adjtimex
[ ]  crond
[ ]  crontab
[*]  dc
[ ]  devfsd
[ ]  eject
[ ]  last
[*]  hdparm
[*]   Support obtaining detailed information directly from drives
[ ]  Register an IDE interface (DANGEROUS)

<Select>  < Exit >  < Help >
```

```

yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Miscellaneous Utilities
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not
[*] Un-register an IDE interface (DANGEROUS)
[*] perform device reset (DANGEROUS)
[*] tristate device for hotswap (DANGEROUS)
[*] get/set using_dma flag (DANGEROUS)
[*] makedevs
    Choose makedevs behaviour (leaf) --->
[*] mt
[*] rx
[*] strings
[*] time
+---+

<Select>  < Exit >  < Help >

```

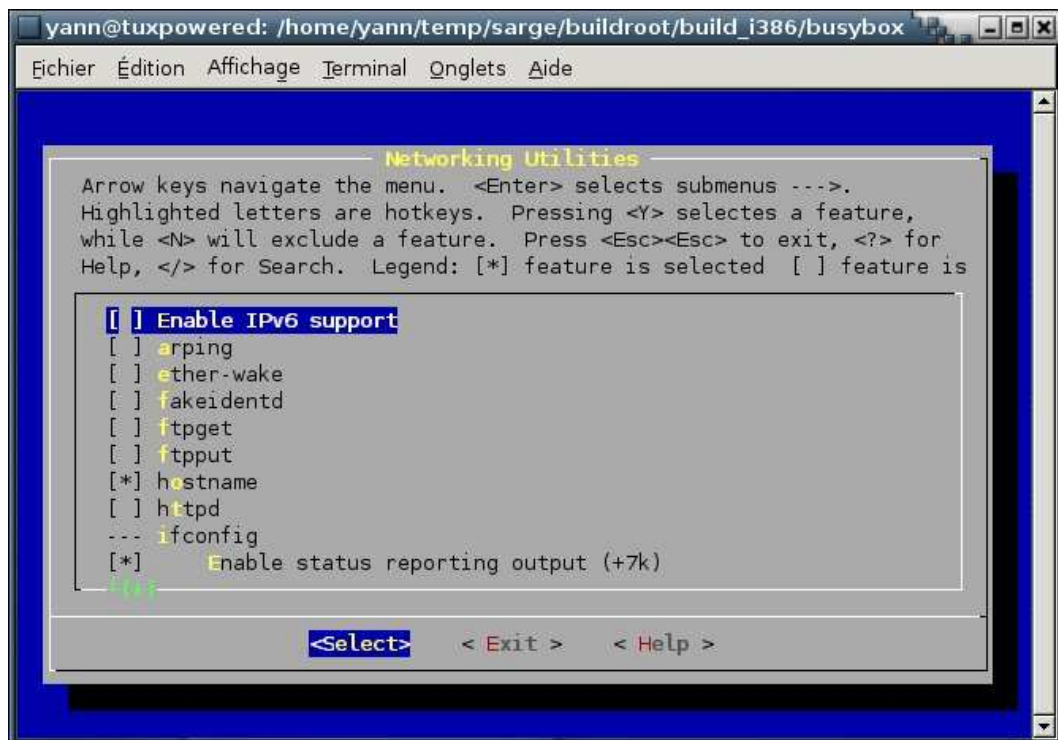
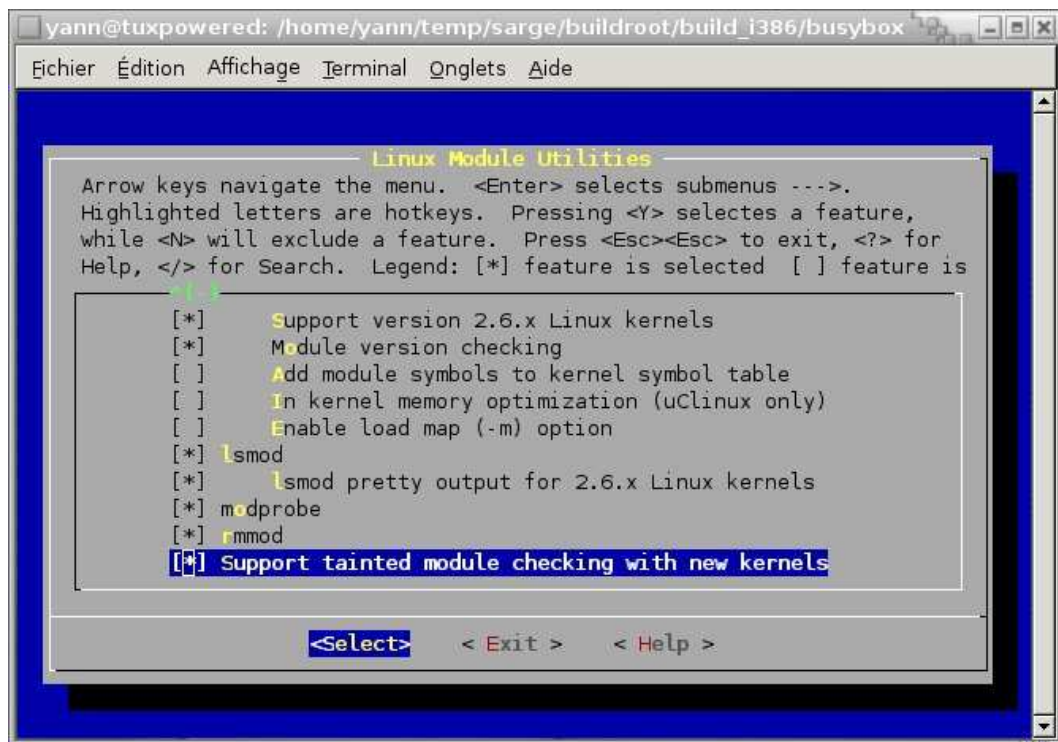
```

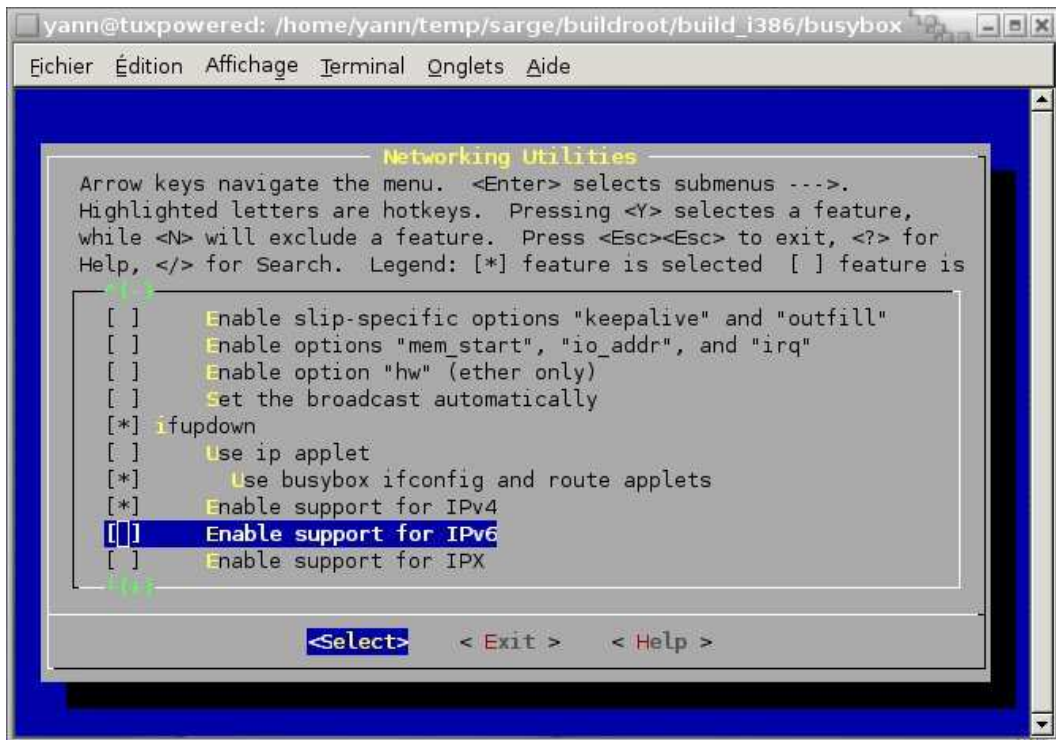
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Linux Module Utilities
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not
[*] insmod
    Support version 2.2.x to 2.4.x Linux kernels
    Support version 2.6.x Linux kernels
    Module version checking
    Add module symbols to kernel symbol table
    In kernel memory optimization (uClinux only)
    Enable load map (-m) option
[*] lsmod
    lsmod pretty output for 2.6.x Linux kernels
[*] modprobe
+---+

<Select>  < Exit >  < Help >

```



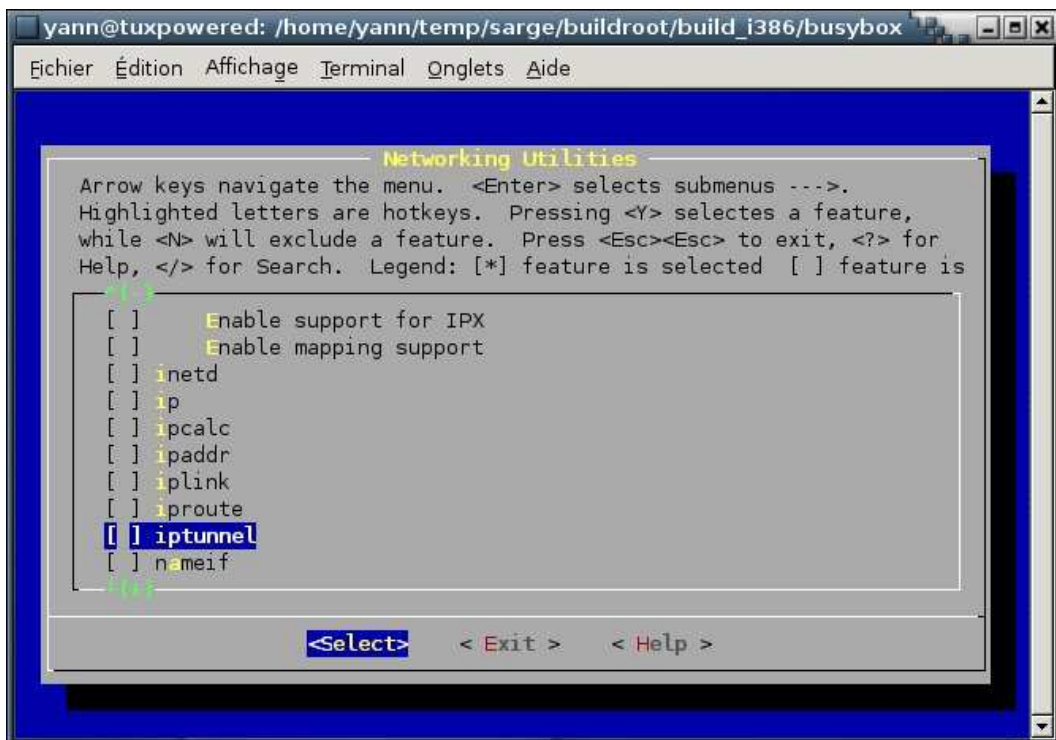


```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Networking Utilities --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not

[*] Enable slip-specific options "keepalive" and "outfill"
[ ] Enable options "mem_start", "io_addr", and "irq"
[ ] Enable option "hw" (ether only)
[ ] Set the broadcast automatically
[*] ifupdown
[ ] Use ip applet
[*] Use busybox ifconfig and route applets
[*] Enable support for IPv4
[ ] Enable support for IPv6
[ ] Enable support for IPX

<Select>  < Exit >  < Help >
```



```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Networking Utilities --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not

[ ] Enable support for IPX
[ ] Enable mapping support
[ ] inetd
[ ] ip
[ ] ipcalc
[ ] ipaddr
[ ] iplink
[ ] iproute
[ ] iptunnel
[ ] nameif

<Select>  < Exit >  < Help >
```

```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

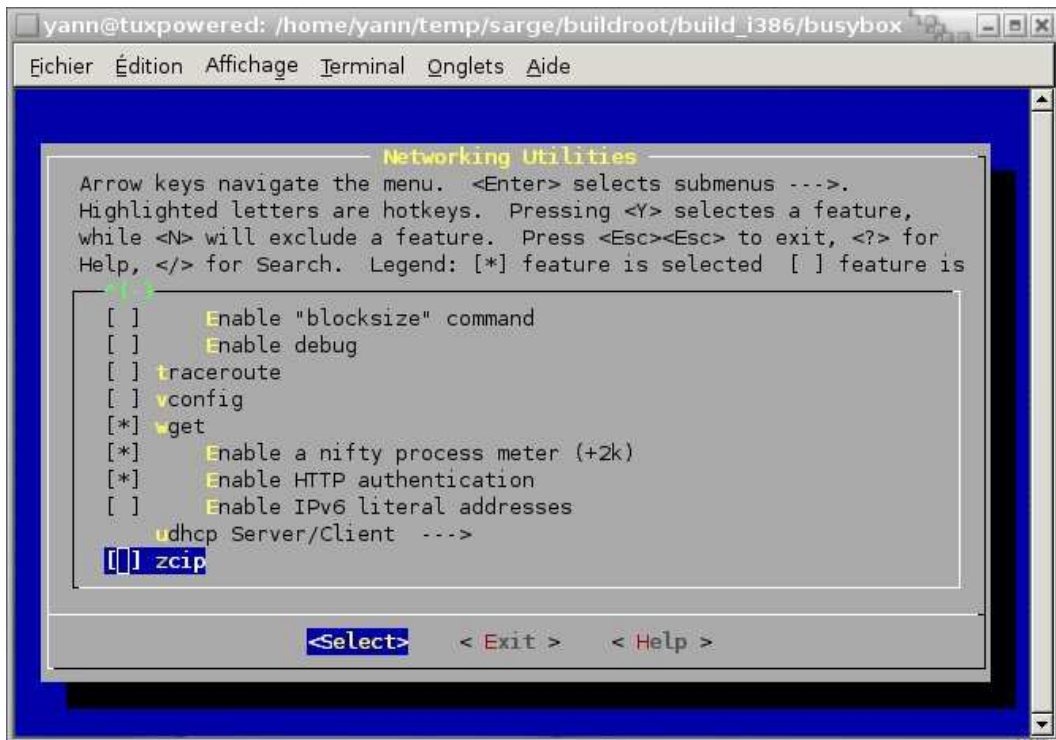
-- Networking Utilities --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not
[*] nc
[*] netstat
[*] nslookup
[*] ping
[*] Enable fancy ping output
--- route
[*] telnet
[*] Pass TERM type to remote host
[ ] Pass USER type to remote host
[ ] telnetd
+---+

<Select>  < Exit >  < Help >
```

```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Networking Utilities --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not
[*] tftp
[*] Enable "get" command
[*] Enable "put" command
[ ] Enable "blocksize" command
[ ] Enable debug
[ ] traceroute
[ ] vconfig
[*] wget
[ ] Enable a nifty process meter (+2k)
[*] Enable HTTP authentication
+---+

<Select>  < Exit >  < Help >
```

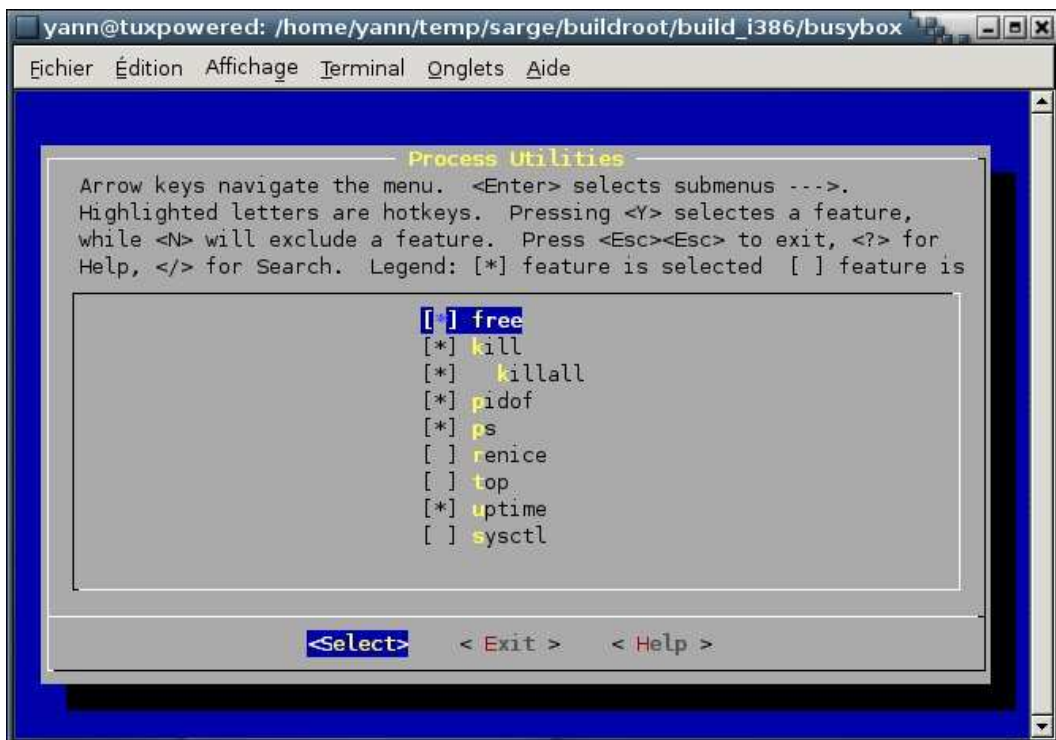



The screenshot shows a terminal window titled "yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox". The menu is titled "Networking Utilities" and contains the following options:

```
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not selected

[ ] Enable "blocksize" command
[ ] Enable debug
[ ] traceroute
[ ] vconfig
[*] wget
[*] Enable a nifty process meter (+2k)
[*] Enable HTTP authentication
[ ] Enable IPv6 literal addresses
[ ] udhcp Server/Client --->
[ ] zcip
```

At the bottom of the menu, there are three buttons: "<Select>", "< Exit >", and "< Help >".

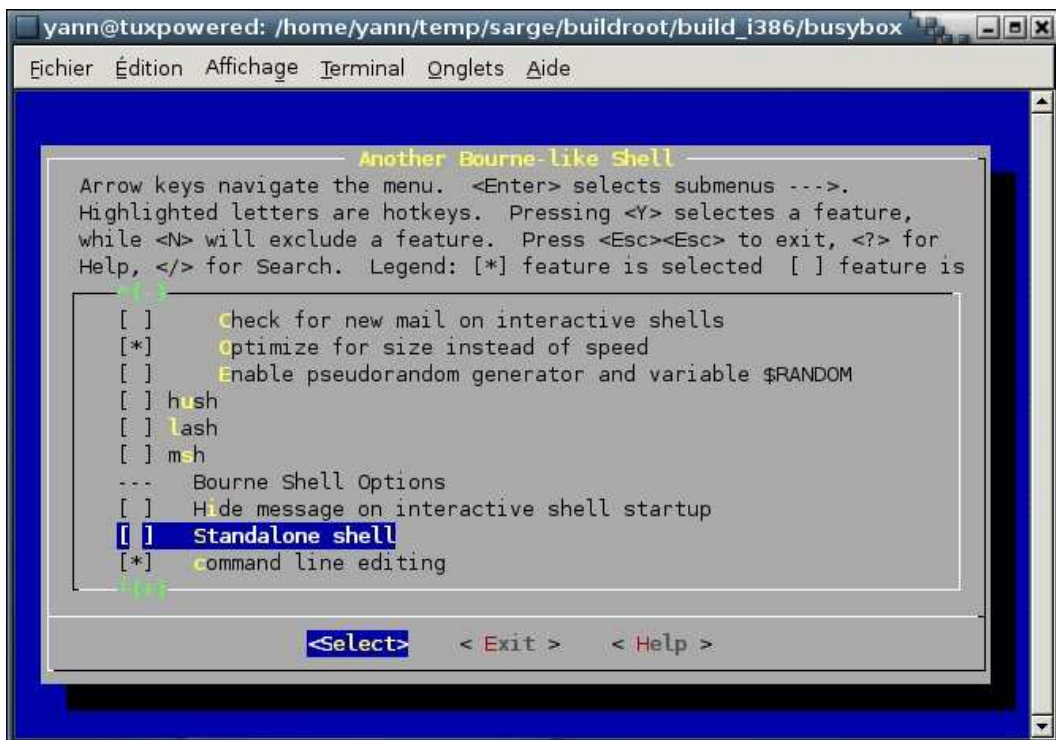
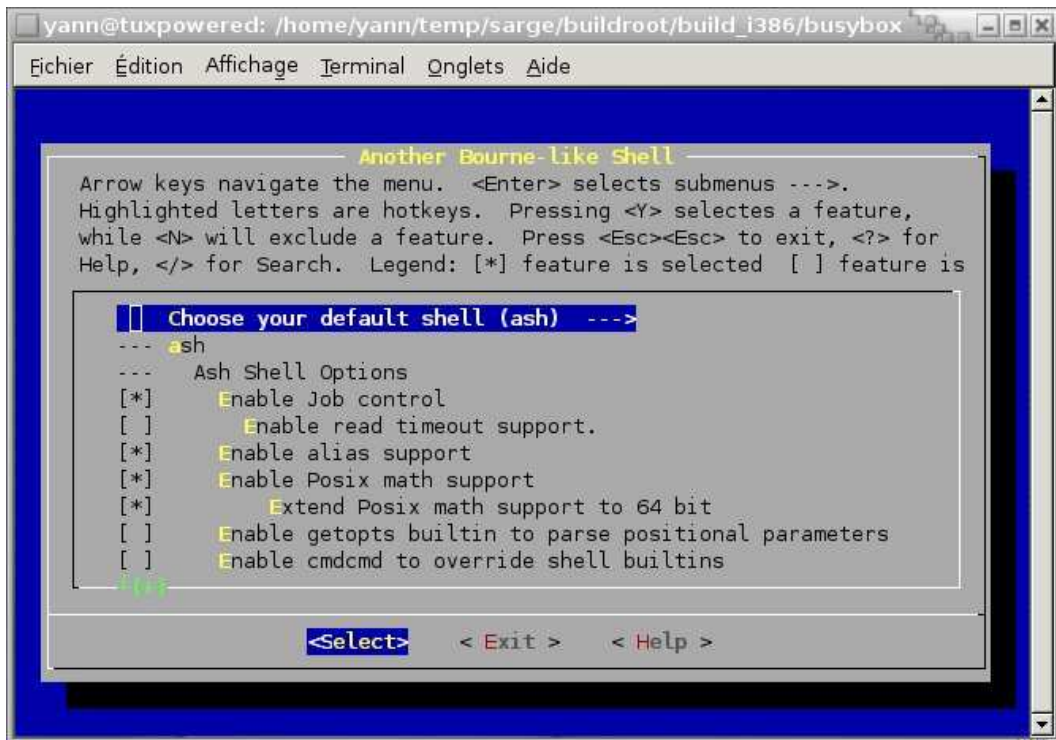


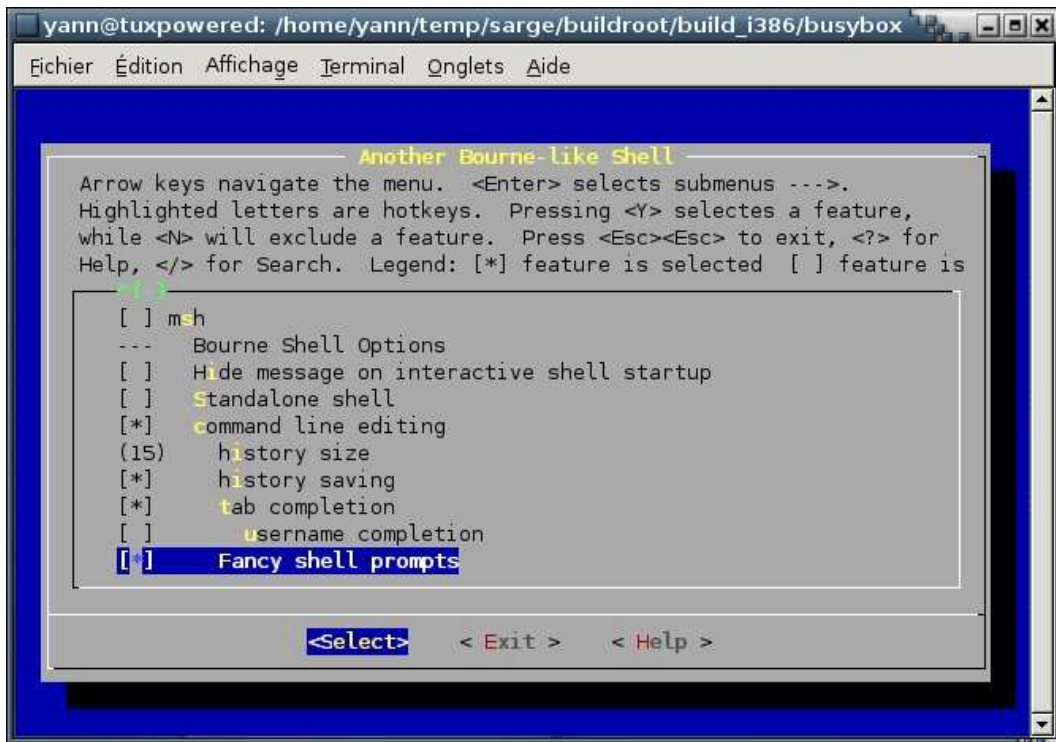
The screenshot shows a terminal window titled "yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox". The menu is titled "Process Utilities" and contains the following options:

```
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not selected

[*] free
[*] kill
[*] killall
[*] pidof
[*] ps
[ ] renice
[ ] top
[*] uptime
[ ] sysctl
```

At the bottom of the menu, there are three buttons: "<Select>", "< Exit >", and "< Help >".





yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox

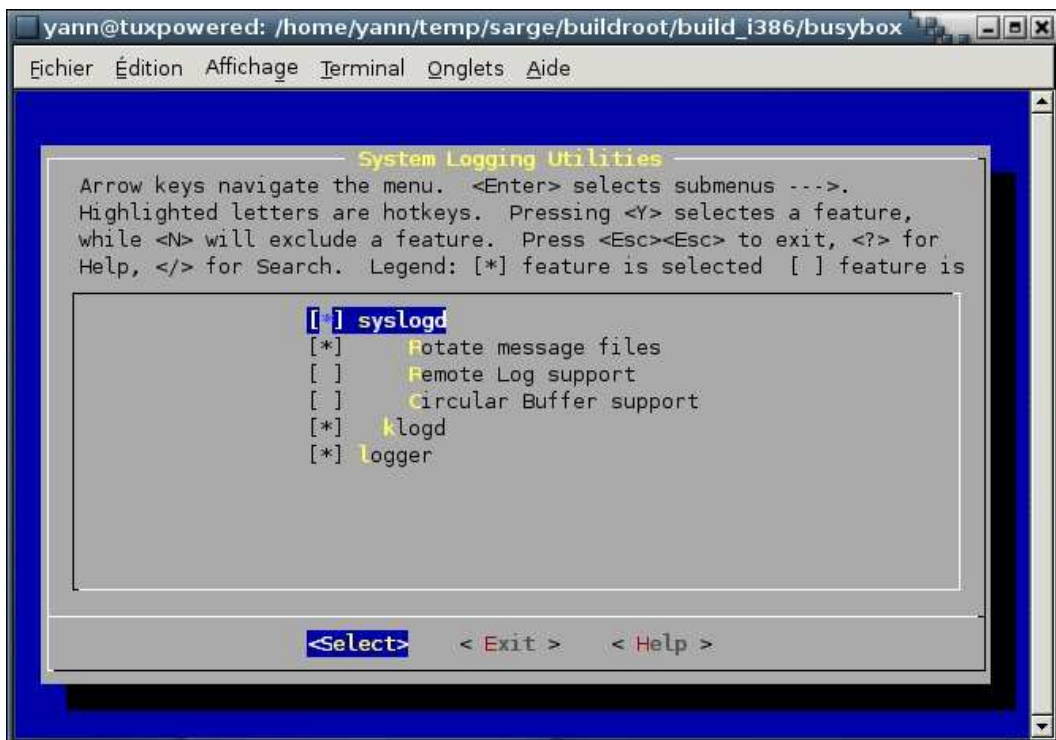
Fichier Édition Affichage Terminal Onglets Aide

— Another Bourne-like Shell —

Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [] feature is

```
[ ] m-h
--- Bourne Shell Options
[ ] Hide message on interactive shell startup
[ ] Standalone shell
[*] command line editing
(15) history size
[*] history saving
[*] tab completion
[ ] username completion
[*] Fancy shell prompts
```

<Select> < Exit > < Help >



yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox

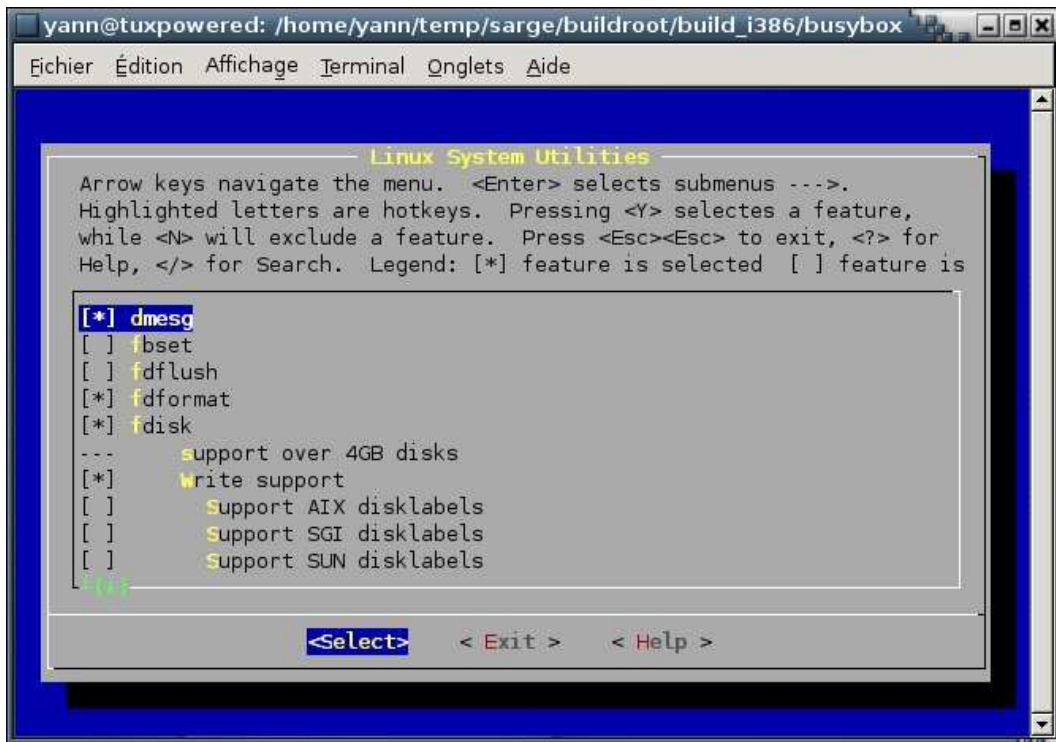
Fichier Édition Affichage Terminal Onglets Aide

— System Logging Utilities —

Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [] feature is

```
[*] syslogd
[*] Rotate message files
[ ] Remote Log support
[ ] Circular Buffer support
[*] klogd
[*] logger
```

<Select> < Exit > < Help >

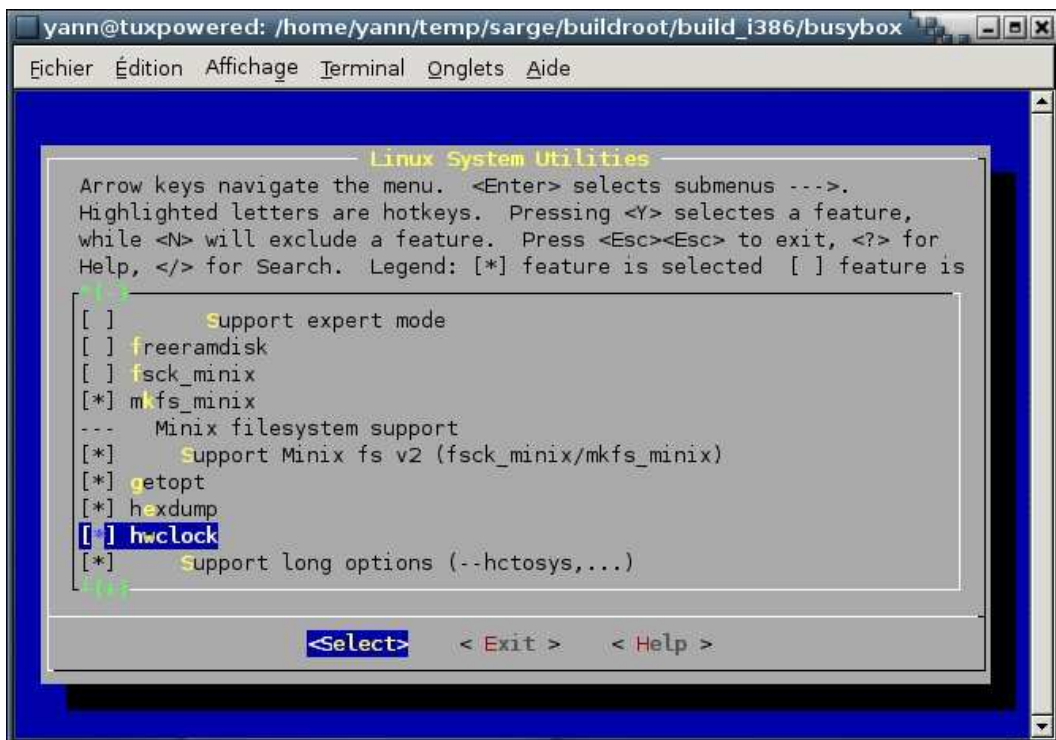


```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Linux System Utilities --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not

[*] dmesg
[ ] fbset
[ ] fdflush
[*] fdformat
[*] fdisk
---      Support over 4GB disks
[*]      Write support
[ ]      Support AIX disklabels
[ ]      Support SGI disklabels
[ ]      Support SUN disklabels
^O^O

<Select>  < Exit >  < Help >
```

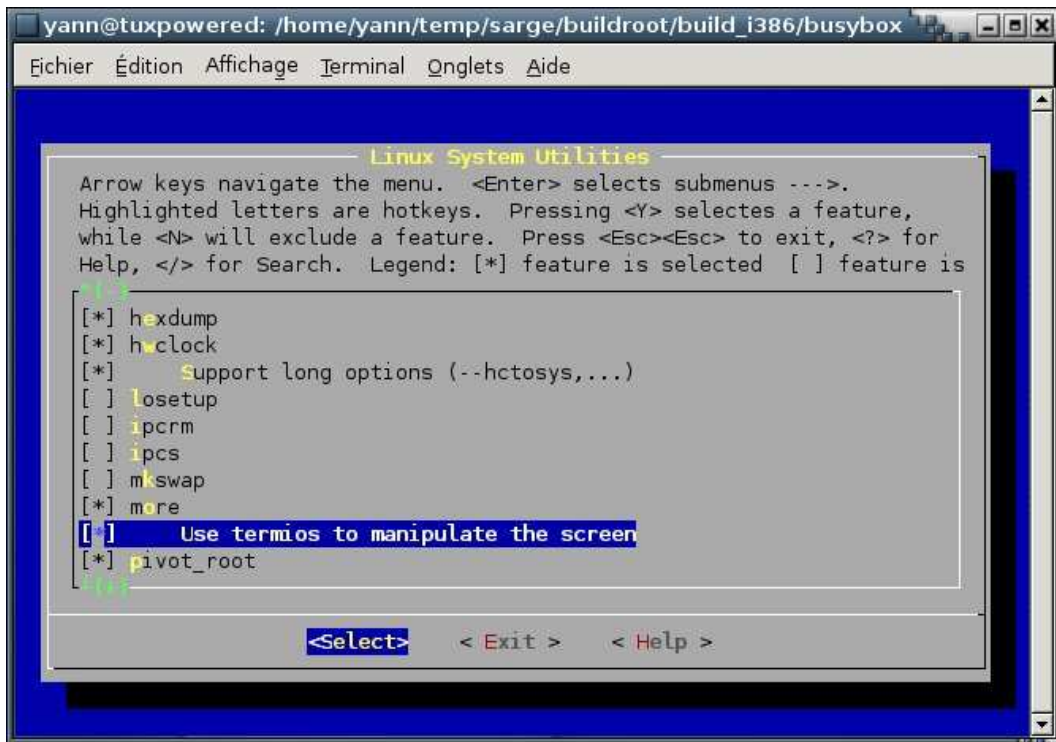


```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

-- Linux System Utilities --
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not

^O^O
[ ]      Support expert mode
[ ] freeramdisk
[ ] fsck_minix
[*] mkfs_minix
---      Minix filesystem support
[*]      Support Minix fs v2 (fsck_minix/mkfs_minix)
[*] getopt
[*] hexdump
[*] hwclock
[*]      Support long options (--hctosys,...)
^O^O

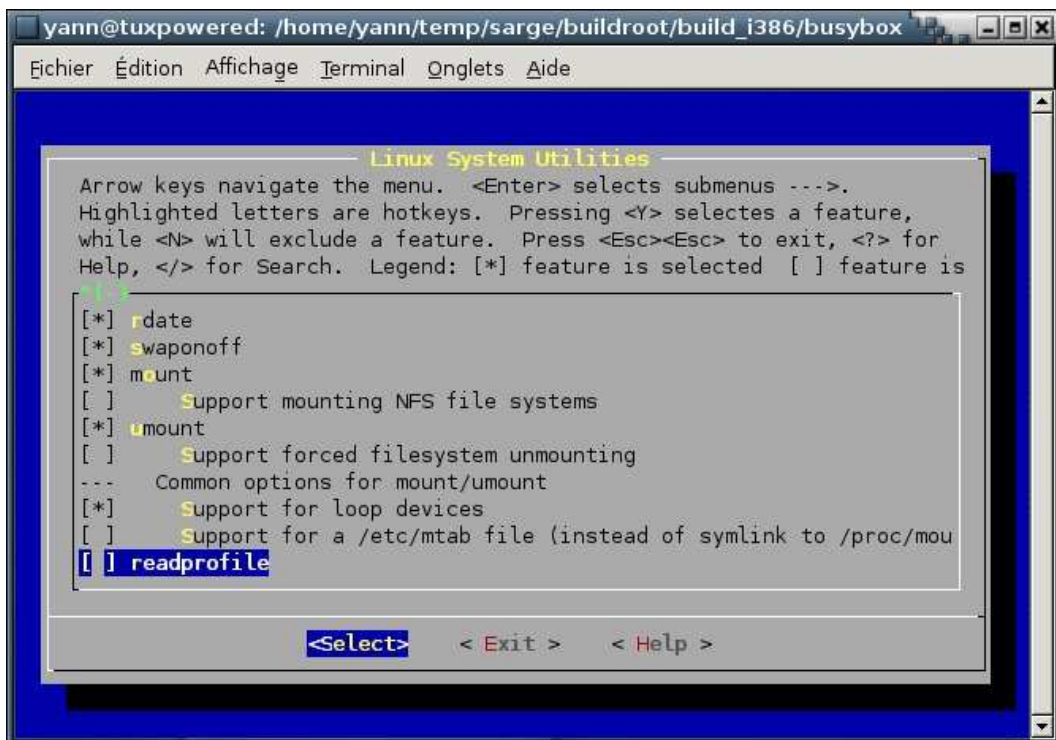
<Select>  < Exit >  < Help >
```

```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Linux System Utilities
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not
[*] hexdump
[*] hwclock
[*] Support long options (--hctosys,...)
[ ] losetup
[ ] ipcrm
[ ] ipcs
[ ] mkswap
[*] more
[+] Use termios to manipulate the screen
[*] pivot_root

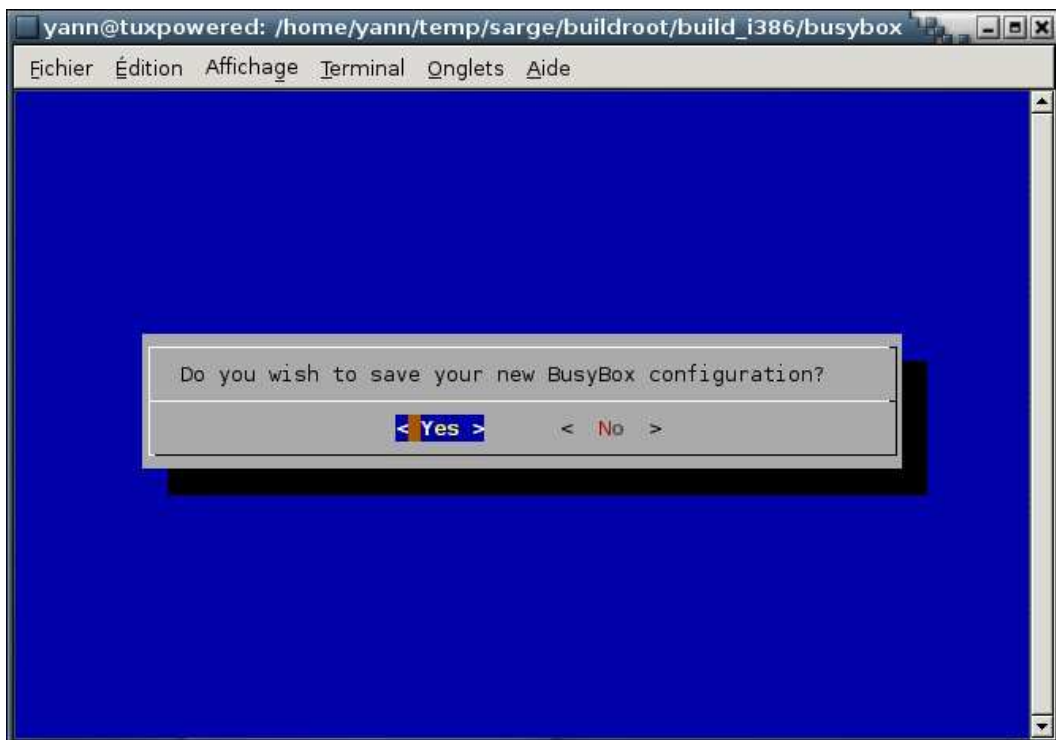
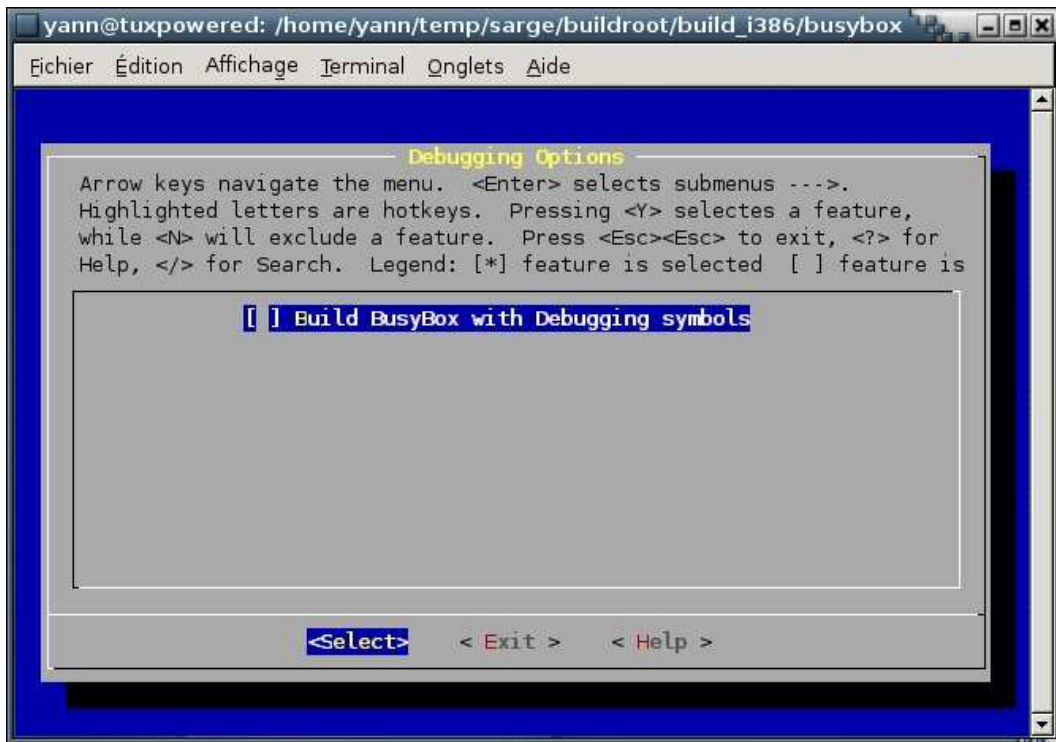
<Select>  < Exit >  < Help >
```

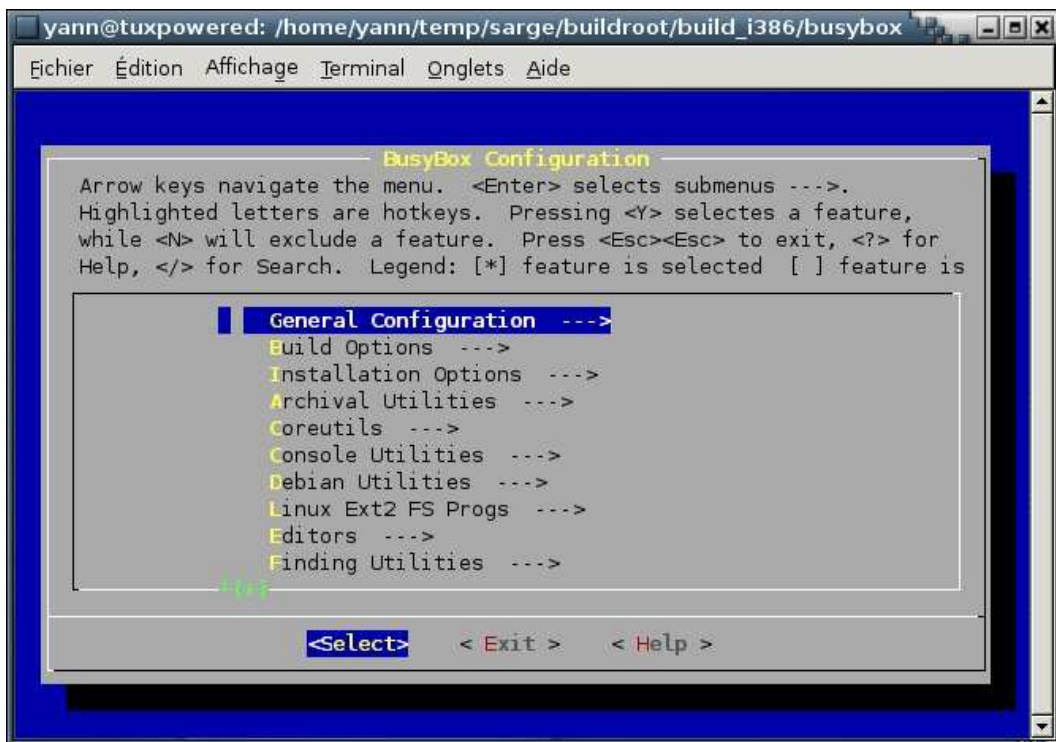
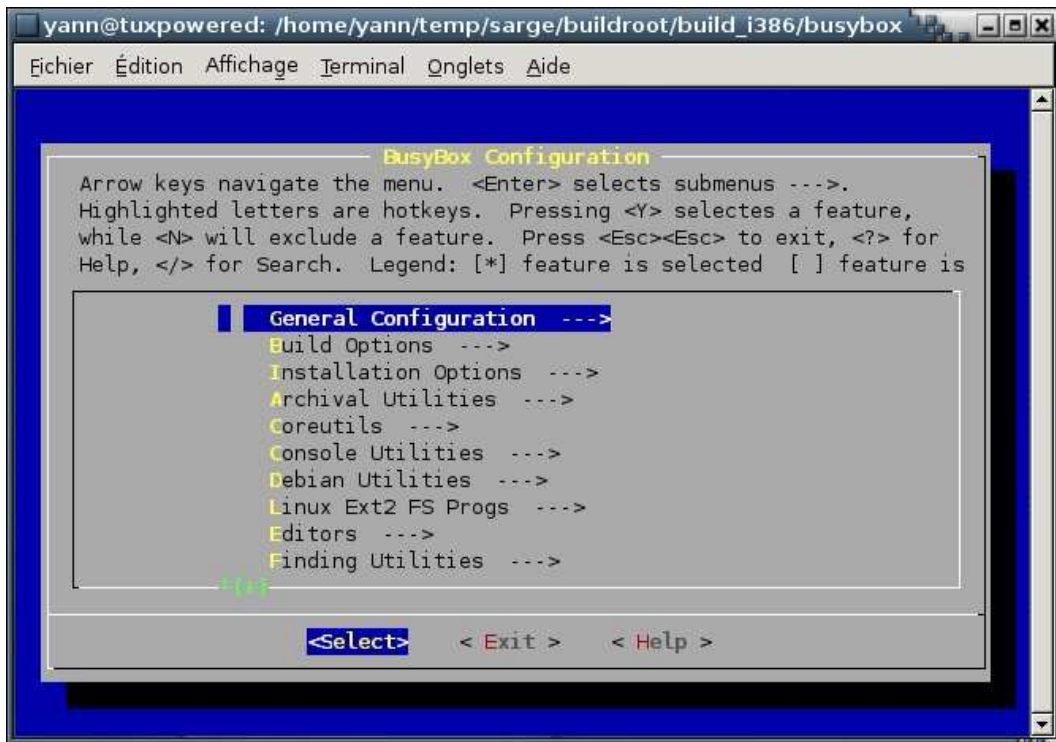


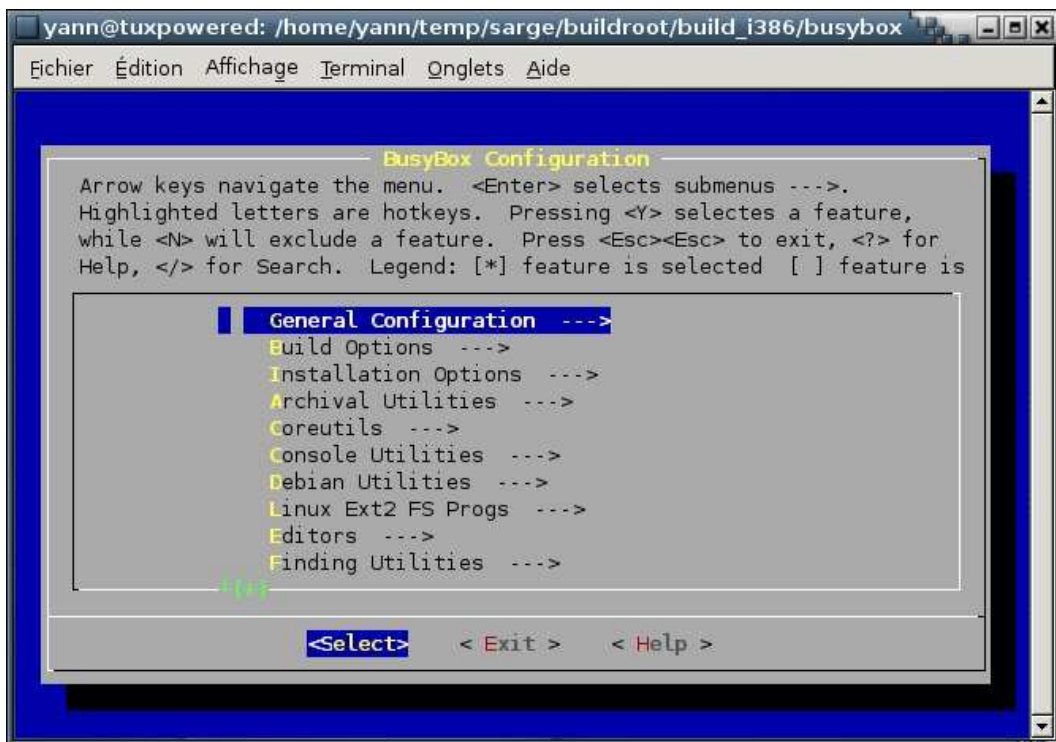
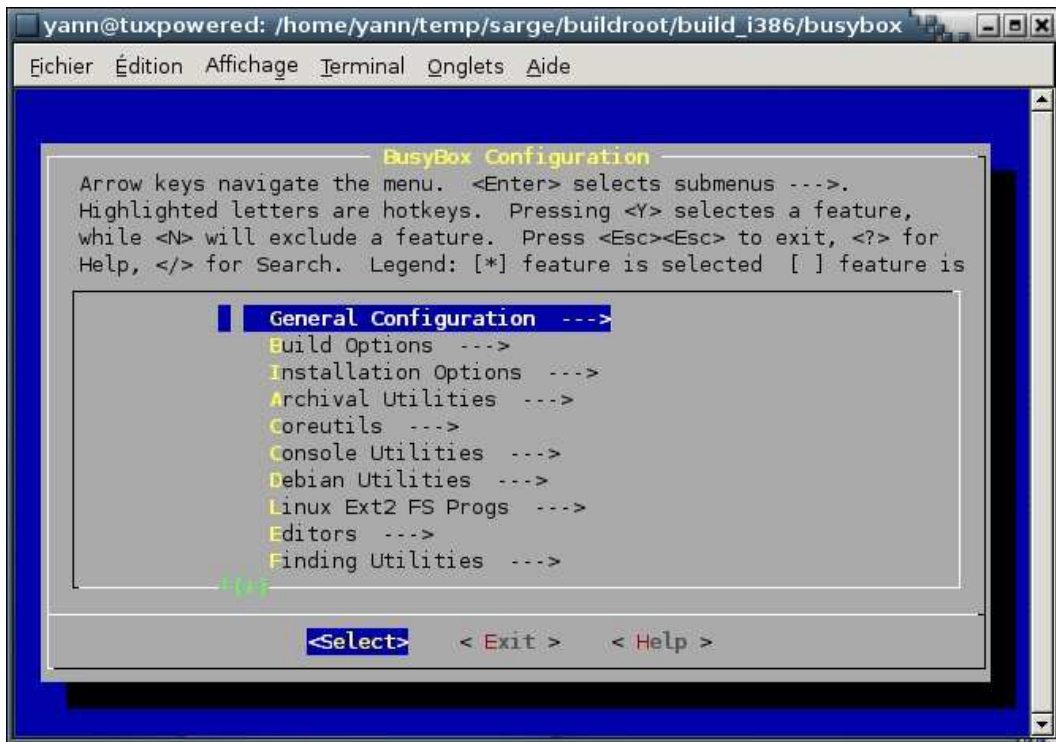
```
yann@tuxpowered: /home/yann/temp/sarge/buildroot/build_i386/busybox
Fichier  Édition  Affichage  Terminal  Onglets  Aide

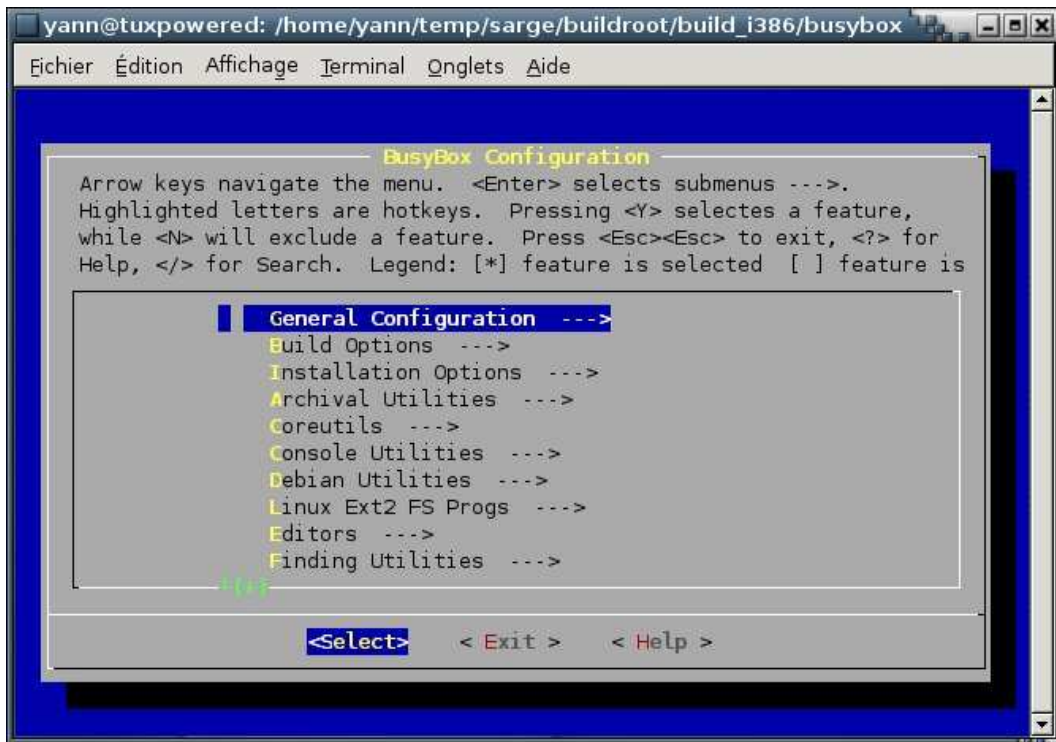
Linux System Utilities
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> selectes a feature,
while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for
Help, </> for Search. Legend: [*] feature is selected [ ] feature is
not
[*] rdate
[*] swaponoff
[*] mount
[ ] Support mounting NFS file systems
[*] umount
[ ] Support forced filesystem unmounting
--- Common options for mount/umount
[*] Support for loop devices
[ ] Support for a /etc/mtab file (instead of symlink to /proc/mou
[ ] readprofile

<Select>  < Exit >  < Help >
```







Voilà la configuration est terminée, on va copier la fichier de configuration au bon endroit et relancer une compilation.

```
yann@tuxpowered:~/temp/sarge/buildroot/build_i386/busybox$ cd ..
yann@tuxpowered:~/temp/sarge/buildroot/build_i386$ cd ..
yann@tuxpowered:~/temp/sarge/buildroot$ cp build_i386/busybox/.config package/busybox/bu
yann@tuxpowered:~/temp/sarge/buildroot$ make
```

Normalement tout doit bien se passer et vous obtenez une image `root_fs_i386.ext2`.

Si vous êtes trop pressé, vous pouvez toujours télécharger mon image [fichier_utils/last/root_fs_i386.ext2.gz](#)

3.3.2 Configuration de notre distribution

Comme je vous l'ai dit plus haut, je dois ajouter quelques programmes dans ce système de fichiers. Il s'agit essentiellement de `udp-cast`, `gpart`, `partimage` qui ont été compilés statiquement dans la distribution BSC. Comme ils fonctionnent bien et que j'ai la place sur la clé USB, je n'ai pas besoin de les recompiler.

- ▷ util-linux fdisk, sfdisk <http://ftp.cwi.nl/aeb/util-linux/>
- ▷ partimage <http://www.partimage.org/index.fr.html>
- ▷ gpart <http://www.stud.uni-hannover.de/user/76201/gpart/>
- ▷ udp-cast mcast <http://udpcast.linux.lu/>
- ▷ testdisk <http://www.cgsecurity.org/index.html?testdisk.html>

Mais pour vous amuser, il est tout a fait possible de le faire. Pour cela vous pouvez utiliser une image d'une système de developpement uClibC pour votre architecture à l'adresse <http://uclibc.org/toolchains.html>.

Ensuite, il ne reste qu'à décompresser l'image, puis la monter dans un répertoire :

```
bunzip2 root_fs_i386.bz2
mkdir root_fs
```

```
su root
mount -o loop root_fs_i386 root_fs
chroot root_fs /bin/su -
```

Ensuite vous pouvez recompiler votre programme en utilisant les outils de votre système embarqué.

En ce qui me concerne je vais monter l'image précédemment créée pour y ajouter quelques fichiers et configurer le réseau et la claviers français.

Pour cela vous aurez besoin des programmes et fichiers suivants

- ▷ `motd.gz` le mot de démarrage que l'on va afficher [fichier_utils/motd.gz](#)
- ▷ `fr.bmap.gz` qui contient la carte du calvier français [fichier_utils/fr.bmap.gz](#)
- ▷ les outils `mcast`, `gpart` etc... [fichier_utils/outils.tgz](#)

```
yann@tuxpowered:~/temp/sarge$ su
Password:
tuxpowered:/home/yann/temp/sarge# mount -o loop root_fs_i386.ext2 root_fs
tuxpowered:/home/yann/temp/sarge#
tuxpowered:/home/yann/temp/sarge# cd root_fs/etc/
tuxpowered:/home/yann/temp/sarge/root_fs/etc# cp /tmp/fr.bmap.gz .
tuxpowered:/home/yann/temp/sarge/root_fs/etc# cp /tmp/motd.gz .
tuxpowered:/home/yann/temp/sarge/root_fs/etc# cd ../sbin/
tuxpowered:/home/yann/temp/sarge/root_fs/sbin# tar xzf /tmp/outils.tgz
```

Voilà les outils sont maintenant dans l'image. Il ne nous reste plus que quelques configurations. Pour obtenir le clavier français au démarrage, on va modifier le fichier `rcS` qui se trouve dans le répertoire `etc/init.d`. On va lui ajouter les deux lignes suivantes :

```
zcat /etc/fr.bmap.gz | loadkmap
zcat /etc/motd.gz
```

Ensuite nous allons créer dans le répertoire `sbin` le fichier `get` suivant.

Fichier `get` (par Yves Agostini) qui permet de télécharger un programme sur le site ftp de l'université.

```
#!/bin/sh
wget "ftp://195.220.226.243/boot/$1.tgz"
tar -zxf "$1.tgz"
rm "$1.tgz"
cat "$1.txt"
```

N'oubliez pas de lui donner les droits d'exécution.

```
chmod 755 get
```

Nous allons terminer par le script qui doit demander la configuration réseau au démarrage. Il s'agit du fichier `network` que l'on va placer dans le répertoire `etc/init.d`. On va juste le renommer `S50network` pour qu'il soit lancé au démarrage. On va aussi désactiver le serveur `ssh`.

```
tuxpowered:/home/yann/temp/sarge/root_fs/etc/init.d# vim S50network
# remplir avec le fichier network ci-dessous
tuxpowered:/home/yann/temp/sarge/root_fs/etc/init.d# mv S50sshd ssh_desactive
tuxpowered:/home/yann/temp/sarge/root_fs/etc/init.d#
```


Fichier `network` (par Yves Agostini) qui permet la configuration du réseau

```
#!/bin/sh

NET="/mnt/fp/net.ini"
TMP="/tmp/tmp.txt"

grep IP $NET > $TMP
IP_DEF='cut -f 2 -d"=" $TMP'
grep MASK $NET > $TMP
MASK_DEF='cut -f 2 -d"=" $TMP'
grep GW $NET > $TMP
GW_DEF='cut -f 2 -d"=" $TMP'

echo -n "IP Address [$IP_DEF] : "
read IP
if [ "$IP" = "" ]; then
  IP=$IP_DEF
fi
echo -n "Netmask [$MASK_DEF] : "
read MASK
if [ "$MASK" = "" ]; then
  MASK=$MASK_DEF
fi
echo -n "Gateway [$GW_DEF] : "
read GW
if [ "$GW" = "" ]; then
  GW=$GW_DEF
fi

echo "IP=$IP" > $NET
echo "MASK=$MASK" >> $NET
echo "GW=$GW" >> $NET

ifconfig eth0 $IP netmask $MASK
route add default gw $GW
```

Il ne reste plus qu'à démonter le système de fichiers

```
tuxpowered:/home/yann/temp/sarge/root_fs/etc/init.d# cd ../../../../
tuxpowered:/home/yann/temp/sarge# umount root_fs
tuxpowered:/home/yann/temp/sarge#
```

On compresse notre image pour la placer sur notre clé USB

```
tuxpowered:/home/yann/temp/sarge# gzip root_fs_i386.ext2
tuxpowered:/home/yann/temp/sarge# mount /dev/sda ramdisk
tuxpowered:/home/yann/temp/sarge# cp root_fs_i386.ext2.gz ramdisk/rootfs.gz
tuxpowered:/home/yann/temp/sarge# umount ramdisk
```

Voilà, tout est presque prêt, il nous reste encore à modifier le fichier `syslinux.cfg` de la distribution debian afin de pouvoir lancer notre mini-distribution.

3.4 Configuration de SYSLinux et test

On y est presque. Le fichier `syslinux.cfg` qui se trouve à la racine de la clé USB, permet de contrôler le fonctionnement du démarrage.

On va donc modifier ce fichier pour y rajouter notre nouvelle mini-distribution.

3.4.1 Explications des différentes parties

Cette première ligne demande à syslinux d'afficher le contenu de `syslinux.txt` ?

```
display syslinux.txt
```

Ce fichier est un fichier texte, qui contient des directives pour l'affichage d'image et de texte en couleur. Pour plus d'informations je vous renvoie à l'adresse suivante <http://syslinux.zytor.com/faq.php>

```
^Xsplash.rle
```

```
Press ^SF1^Tcontrol and F then 1^W for help, or ENTER to
```

La ligne suivante décrit l'action par défaut. Comme pour Lilo, il s'agit d'un label qui pointe sur la description d'un noyau, d'un système de fichiers racine et d'options.

```
default linux
```

Ces lignes permettent d'afficher le fichier texte correspondant lors de l'appui de la touche. L'appui sur la touche F1 provoque l'affichage de F1.txt qui peut être formater de manière à contenir des images et des couleurs.

```
F1 f1.txt
F2 f2.txt
F3 f3.txt
F4 f4.txt
F5 f5.txt
F6 f6.txt
F7 f7.txt
F8 f8.txt
F9 f9.txt
F0 f10.txt
```

et enfin les lignes concernant quels noyaux il est possible de démarrer.

```
label linux
    kernel linux
    append vga=normal initrd=initrd.gz ramdisk_size=9650 root=/dev/rd/0
    devfs=mount,dall rw --
label expert
    kernel linux
    append DEBCONF_PRIORITY=low vga=normal initrd=initrd.gz
    ramdisk_size=9650 root=/dev/rd/0 devfs=mount,dall rw --
label linux26
    kernel linux26
```

```
        append vga=normal initrd=initrd26.gz ramdisk_size=10396 root=/dev/rd/0 devfs=mou
label expert26
        kernel linux26
        append DEBCONF_PRIORITY=low vga=normal initrd=initrd26.gz
        ramdisk_size=10396 root=/dev/rd/0 devfs=mount,dall rw  --
```

C'est dans cette partie que nous rajouterons les lignes suivantes qui nous permettrons de démarrer sur notre distribution.

```
label rescue
        kernel linuzrs
        append vga=normal initrd=rootfs.gz ramdisk_size=40960
root=/dev/ram0 rw
```

Notre fichier syslinux final ressemble à ceci :

```
display syslinux.txt
default linux
```

```
F1 f1.txt
F2 f2.txt
F3 f3.txt
F4 f4.txt
F5 f5.txt
F6 f6.txt
F7 f7.txt
F8 f8.txt
F9 f9.txt
F0 f10.txt
```

```
label linux
        kernel linux
        append vga=normal initrd=initrd.gz ramdisk_size=9650 root=/dev/rd/0
        devfs=mount,dall rw  --
label expert
        kernel linux
        append DEBCONF_PRIORITY=low vga=normal initrd=initrd.gz
        ramdisk_size=9650 root=/dev/rd/0 devfs=mount,dall rw  --
label linux26
        kernel linux26
        append vga=normal initrd=initrd26.gz ramdisk_size=10396 root=/dev/rd/0 devfs=mou
label expert26
        kernel linux26
        append DEBCONF_PRIORITY=low vga=normal initrd=initrd26.gz
        ramdisk_size=10396 root=/dev/rd/0 devfs=mount,dall rw  --

label rescue
        kernel linuzrs
        append vga=normal initrd=rootfs.gz ramdisk_size=40960
root=/dev/ram0 rw
```

```
prompt 1
timeout 0
```

3.4.2 Bonus : modification de l'image splash.rle

Comme indiqué à l'adresse <http://syslinux.zytor.com/faq.php>, le format de l'image `splash.rle` est de type LSS16 qui est une image 640x480 en mode 16 couleurs.

Pour la modifier, nous avons besoin de la convertir en un format lisible par `gimp` par exemple.

Pour cela on utilise le programme `ppmtolss16` :

```
yann@tuxpowered:~/temp/sarge/cle64mo$ lss16toppm < splash.rle > splash.ppm
```

on peut ensuite éditer l'image avec `gimp`



Puis la modifier



Ensuite ne pas oublier d'enregistrer en mode 16 couleurs au format ppm. Il ne reste plus qu'à reconvertir en fichier LSS16 par la commande :

```
yann@tuxpowered:~/temp/sarge/cle64mo$ ppmtolss16 < splash.ppm > splash.rle
```

puis copier le fichier modifier sur la clé USB.

Voilà tout est finalement prêt. On peut maintenant passer aux tests

3.4.3 Tests

Après avoir branché la clé, on peut alors tester notre distribution sur clé USB après avoir fait une image par la commande :

```
yann@tuxpowered:~/temp/sarge$ dd if=/dev/sda of=cle64mo.img
```

On peut aussi tester cette image à l'aide QEMU <http://fabrice.bellard.free.fr/qemu/>. Les tests ont aussi été faits sur une petite divxbox à base de carte EPIA M10000. Tout fonctionne bien.

On commence par installer QEMU.

```
apt-get install qemu
```

La version sarge est la 0.61 qui fera très bien l'affaire pour notre test.

On va donc lancer QEMU sur l'image de notre clé.

```
yann@tuxpowered:~/temp/sarge$ qemu cle64mo.img
```

ce qui donne :



La suite de la séquence de boote est identique à la suite.

On peut aussi tester le système de fichier et le noyau directement.

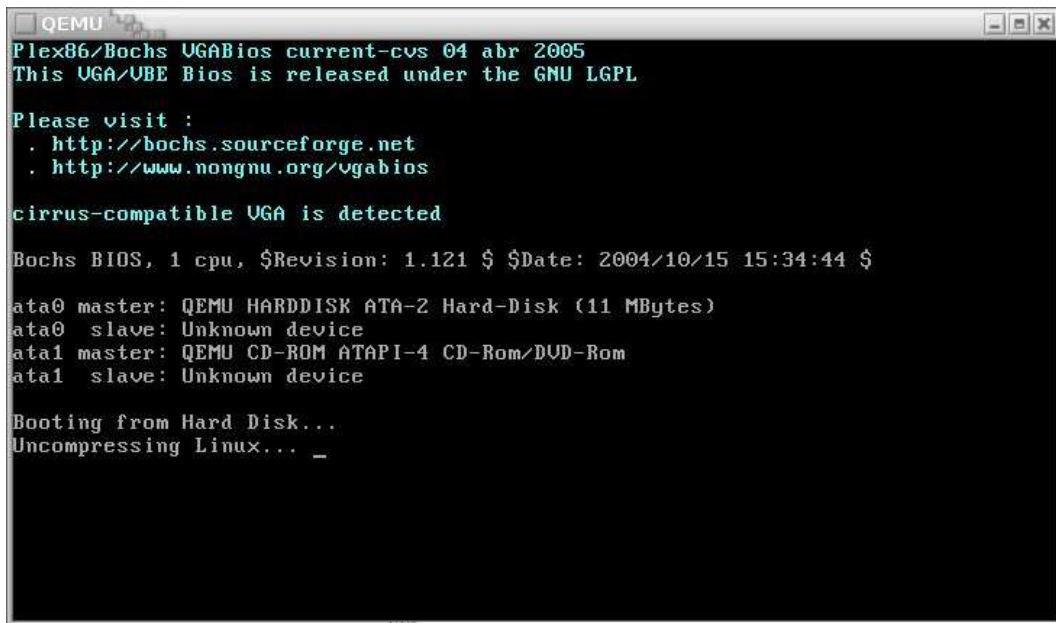
Il faut décompresser l'image du système de fichier que l'on a créé.

```
yann@tuxpowered:~/temp/sarge/cle64mo$ gunzip rootfs.gz
```

ensuite il ne reste plus qu'à invoquer `qemu` en s'inspirant de l'exemple fournit sur la site <http://fabrice.bellard.free.fr/qemu/download.html>


```
qemu -hda rootfs -kernel linuzrs -append "root=/dev/hda
ide2=noprobe ide3=noprobe ide4=noprobe ide5=noprobe"
```

et l'on obtient en images :



```
QEMU
Plex86/Bochs VGABios current-cvs 04 abr 2005
This VGA/UBE Bios is released under the GNU LGPL

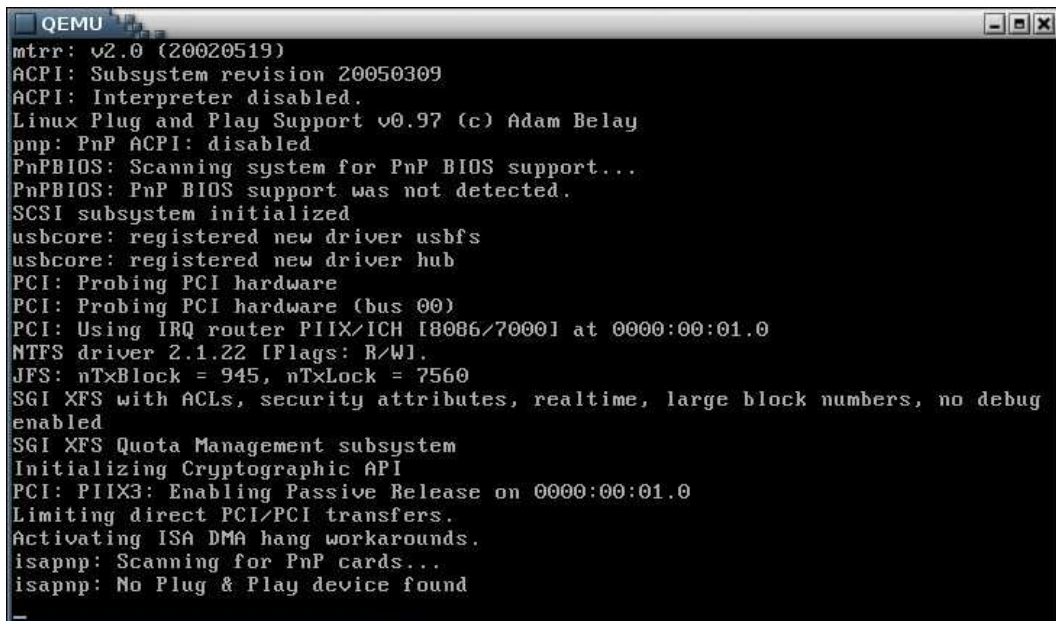
Please visit :
 . http://bochs.sourceforge.net
 . http://www.nongnu.org/vgabios

cirrus-compatible VGA is detected

Bochs BIOS, 1 cpu, $Revision: 1.121 $ $Date: 2004/10/15 15:34:44 $

ata0 master: QEMU HARDDISK ATA-2 Hard-Disk (11 MBytes)
ata0 slave: Unknown device
ata1 master: QEMU CD-ROM ATAPI-4 CD-Rom/DVD-Rom
ata1 slave: Unknown device

Booting from Hard Disk...
Uncompressing Linux... _
```



```
QEMU
mtrr: v2.0 (20020519)
ACPI: Subsystem revision 20050309
ACPI: Interpreter disabled.
Linux Plug and Play Support v0.97 (c) Adam Belay
pnp: PnP ACPI: disabled
PnPBIOS: Scanning system for PnP BIOS support...
PnPBIOS: PnP BIOS support was not detected.
SCSI subsystem initialized
usbcore: registered new driver usbfs
usbcore: registered new driver hub
PCI: Probing PCI hardware
PCI: Probing PCI hardware (bus 00)
PCI: Using IRQ router PIIX/ICH [8086/7000] at 0000:00:01.0
NTFS driver 2.1.22 [Flags: R/W]
JFS: nTxBlock = 945, nTxLock = 7560
SGI XFS with ACLs, security attributes, realtime, large block numbers, no debug
enabled
SGI XFS Quota Management subsystem
Initializing Cryptographic API
PCI: PIIX3: Enabling Passive Release on 0000:00:01.0
Limiting direct PCI/PCI transfers.
Activating ISA DMA hang workarounds.
isapnp: Scanning for PnP cards...
isapnp: No Plug & Play device found
```

```

QEMU
NET: Registered protocol family 8
NET: Registered protocol family 20
input: ImExPS/2 Generic Explorer Mouse on isa0060/serio1
md: Autodetecting RAID arrays.
md: autorun ...
md: ... autorun DONE.
UFS: Mounted root (ext2 filesystem) readonly.
Freeing unused kernel memory: 400k freed
EXT2-fs warning: mounting unchecked fs, running e2fsck is recommended

***** Disquette Systeme *****
http://www.crium.univ-metz.fr/          crium-reseau@univ-metz.fr
                Version 1.0 13 / 02 / 2002
Disquette de boot pour gerer des PC sans installer linux sur la machine.
*****
- USAGE :
1. Configurez vos informations réseau pour cette machine
   Utilisez une adresse IP disponible.
2. Tapez "get package"
   La liste des packages disponibles se trouve sur
   ftp://ftp.univ-metz.fr/boot
3. "halt", "reboot" ou "Ctrl+Alt+Suppr" pour rebooter.
--

Populating /dev using udev...

```

```

QEMU
- USAGE :
1. Configurez vos informations réseau pour cette machine
   Utilisez une adresse IP disponible.
2. Tapez "get package"
   La liste des packages disponibles se trouve sur
   ftp://ftp.univ-metz.fr/boot
3. "halt", "reboot" ou "Ctrl+Alt+Suppr" pour rebooter.
--

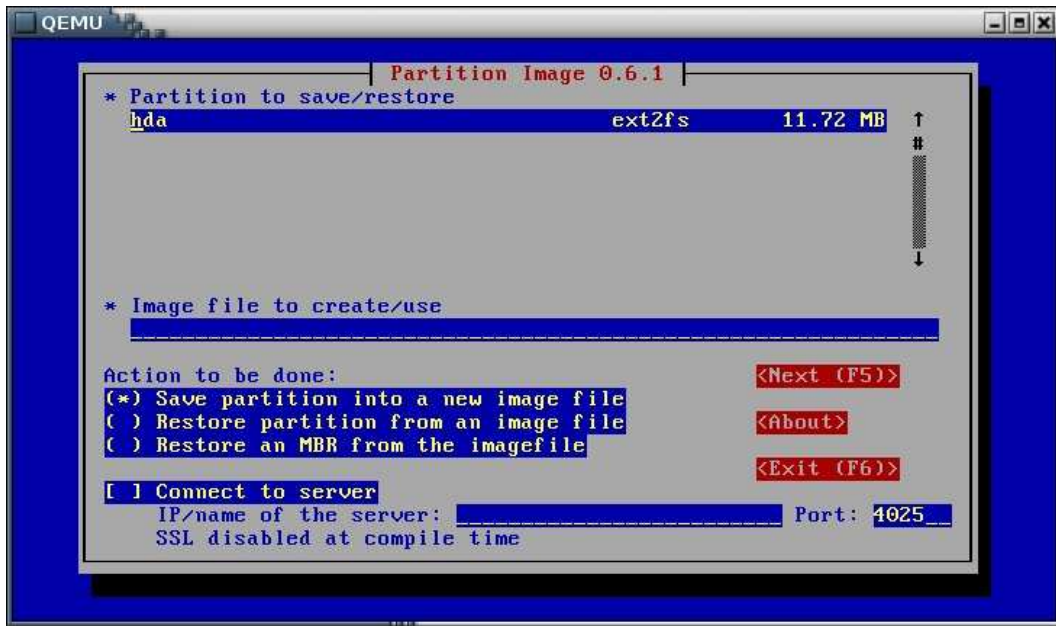
Populating /dev using udev... done
Initializing random number generator... rm: unable to stat '/etc/random-seed': I
nput/output error
urandom start: failed.
done.
Starting network...
IP Address [194.57.140.145] :
Netmask [255.255.255.0] :
Gateway [194.57.140.254] :
uclibc login: root

BusyBox v1.00 (2005.07.13-17:18+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

#

```

On peut même lancer partimage, pour voir...



4 Création et utilisation de la clé USB

4.1 Création de la clé

Dans un premier temps, téléchargez l'image de la clé usb sur cette même page. Décompressez là, insérez votre clé USB (attention toutes les informations se trouvant sur la clé seront détruites).

La taille de la clé importe peu, il suffit que sa taille soit d'au moins de 64Mo.

Dans mon cas la clé se trouve sur le périphérique `/dev/sda`, faites attention, vérifiez bien le périphérique de votre clé USB, car si vous avez des disque SATA/SCSI, vous risqueriez, avec cet commande, d'écraser votre disque principal.

On va ensuite copier l'image de la clé. L'enchaînement des commandes est le suivant :

```
yoda:~# tar xzf cle64.img.gz
yoda:~# dd if=cle64.img of=/dev/sda
```

La clé est maintenant opérationnelle. Il ne reste plus qu'à choisir de booter sur la clé USB dans votre BIOS, pour cela je vous renvoie à la documentation de votre carte mère.

La machine reboote, et vous obtenez ce qui suit :



Vous avez alors la possibilité :

- ▷ booter sur l'installation de Debian Sarge (noyau 2.4 et noyau 2.6)
- ▷ booter sur la mini-distribution de dépannage/clonage/restauration

Installation Debian Sarge

- ▷ A l'invite de commande, si vous taper simplement `enter`, vous booter sur un noyau 2.4 pour l'installation Debian Sarge.
- ▷ A l'invite de commande, si vous taper `linux26`, vous booter sur un noyau 2.6 (obligatoire dans le cas de disque SATA) pour l'installation Debian Sarge.

Mini-distribution dépannage/clonage/restauration A l'invite de commande, si vous taper `rescue`, vous booter sur un gros noyau 2.6 monolithique intégrant un maximum de pilotes de périphériques.

A la fin du démarrage, un script vous demande de configurer le réseau (il peut éventuellement être masqué par des messages d'erreur dus à la clé USB). Dans tous les cas, lorsque vous devez rentrer une configuration réseau en commençant par l'adresse IP de la machine.

```

QEMU
- USAGE :
1. Configurez vos informations réseau pour cette machine
   Utilisez une adresse IP disponible.
2. Tapez "get package"
   La liste des packages disponibles se trouve sur
      ftp://ftp.univ-metz.fr/boot
3. "halt", "reboot" ou "Ctrl+Alt+Suppr" pour rebooter.
--

Populating /dev using udev... done
Initializing random number generator... rm: unable to stat '/etc/random-seed': I
nput/output error
urandom start: failed.
done.
Starting network...
IP Address [194.57.140.145] :
Netmask [255.255.255.0] :
Gateway [194.57.140.254] :
uclibc login: root

BusyBox v1.00 (2005.07.13-17:18+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

# _

```

ensuite à l'invite `uclibc login`, taper `root` puis `enter`.

Vous pouvez maintenant commencer à utiliser les outils mis à votre disposition :

- ▷ util-linux `fdisk`, `sfdisk` <http://ftp.cwi.nl/aeb/util-linux/>
- ▷ partitionnement <http://www.partition.org/index.fr.html>
- ▷ `gpart` <http://www.stud.uni-hannover.de/user/76201/gpart/>
- ▷ `udp-cast` `mcast` <http://udpcast.linux.lu/>
- ▷ `testdisk` <http://www.cgsecurity.org/index.html?testdisk.html>

4.2 Clonage de machine

Dans un premier temps connectez les machines à cloner ainsi que la master sur un switch ou un hub. Dans le cas d'un hub, ne pas connecter le `uplink` vers le réseau externe de la salle à cloner. En effet cela aurait pour effet de propager les données de clonage sur les autres réseaux, ce qui ralentirait très fortement toutes les autres communications. Le conseil est donc d'isoler la salle à cloner.

Démarrez toutes les machines sur la mini-distribution `rescue` et donnez à chacune une adresse IP différente (par exemple dans la classe 192.168.1.).

L'utilitaire est `udpcast` va envoyer sur le réseau local en UDP l'image du disque dur du master, et les machines à cloner vont écouter et récupérer ce qui passe sur le réseau pour l'écrire sur leur propre disque.

Le réseau local est alors saturé par cet fonction et 2 cas sont possibles.

Le premier utilise la compression, les commandes sont alors les suivantes :

- ▷ `udp-sender -p "gzip -c" -f /dev/hda` pour la machine *Master* (celle qui émet).
- ▷ `udp-receiver -p "gzip -dc" -f /dev/hda` pour la machine à cloner (celle qui reçoit les données).

Le second test n'utilise pas la compression, les commandes sont les suivantes :

- ▷ `udp-sender -f /dev/hda` pour la machine *Master* (celle qui émet).
- ▷ `udp-receiver -f /dev/hda` pour la machine à cloner (celle qui reçoit les données).

Sur un réseau 100MB avec des cartes 100MB, un disque dur de 80Go est cloné en 2h30.

5 Crédits

C'en est fini de cet article, toutes remarques et corrections sont les bienvenues à l'adresse morere@univ-metz.fr