

BAB 13

LINUX DI FLOPPY DRIVE

Salah satu alasan orang untuk menggunakan aplikasi opensource seperti GNU/Linux karena dapat dikustomasi sesuai dengan keinginan penggunanya. Banyak aplikasi yang dapat kita buat dengan berbagai macam software open source yang tersedia di internet.

Pada bab ini penulis mencoba menuangkan salah satu kustomasi GNU/linux yakni pembuatan distribusi GNU/linux pada floppy drive berbasis debian GNU/Linux.

Penggunaan aplikasi ini banyak digunakan untuk keperluan seperti proyek embedded GNU/linux (embedded multimedia, embedded printer, dll), pembuatan router, medium instalasi selain penggunaan media CD, Hardisk, USB serta aplikasi-aplikasi lain yang dapat anda buat sendiri.

Untuk pembuatan distribusi GNU/Linux pada floppy drive, ada beberapa hal yang perlu diperhatikan seperti kapasitas floppy drive, software-software pendukung yang nantinya ditancapkan ke dalam floppy drive. Dengan pembuatan distribusi ini, kita harus menghemat resource software yang akan disertakan mengingat kapasitas floppy drive yang sangat terbatas.

Paket Software Yang Dibutuhkan

a. Kernel linux

penulis menggunakan kernel linux bawaan distribusi Debian Woody, yakni kernel-source-2.4.18.deb yang berada pada CD #1 installer Debian Woody.

b.uClibc-0.9.13.tar.gz

Software ini merupakan librari C khusus untuk keperluan pengembangan sistem embedded GNU/Linux yang dibuat oleh Erik Andersen (andersen@codepoet.org). Kapasitas dari software ini lebih kecil dibanding librari GNU C yang dipakai pada distribusi-distribusi GNU/linux yang ada saat ini.

Selain itu, uClibc mendukung berbagai jenis arsitektur komputer seperti alpha, ARM, i386, i960, h8300, m68k, mips/mipsel, PowerPC, SH, SPARC, serta v850.

c. busybox-1.00-pre2.tar.gz

Seperti halnya uClibc, software ini khusus dirancang untuk keperluan pengembangan sistem embedded GNU/linux. Software ini dirancang dengan pembatasan resource dan optimalisasi kapasitas software. Busybox merupakan utilitas UNIX yang dapat dijumpai pada *core* GNU, dan utilitas linux.

d.Boot loader lilo_22.2-3_i386.deb

Paket software ini juga didapatkan pada CD #1 installer Debian Woody ('/pool/main/l/lilo/').

Persiapan Awal

1. Langkah pertama yang dilakukan adalah pembuatan sub-direktori 'myboot' pada path linux anda (Misal: /home/kari/myboot)

```
debian:~# mkdir myboot
debian:~# ls
Desktop           data presentasi
MyMusic          kde
OpenOffice.org1.1.0  kumpulan cerpen
260
Debian GNU/Linux 2nd Edition
Askari Azikin
©2004-2007, http://www.debianindonesia.org
E-mail: kari@debianindonesia.org
```

```
aplikasi          linux on floppy
buku debian jilid 2    linux on floppy.doc
bukubaru        myboot
slidea~1.sxi
```

2. Buat direktori 'uclibc-dev' dan 'rootfs' di bawah direktori myboot

```
debian:~# cd myboot
debian:/myboot# mkdir rootfs uclibc-dev
```

3. Salin source 'Busybox-1.00-pre2.tar.gz' dan 'uClibc-0.9.13.tar.gz' ke direktori myboot

```
debian:/myboot# ls
busybox-1.00-pre2.tar.gz
uClibc-0.9.13.tar.gz
rootfs
uclibc-dev
```

4. Ekstrak seluruh source tersebut dengan perintah berikut:

```
debian:/myboot# tar zxvf uClibc-0.9.13.tar.gz
debian:/myboot# tar zxvf busybox-1pre2.tar.gz
atau
debian:/myboot# gunzip -c uClibc-0.9.13.tar.gz | tar xvf -
debian:/myboot# gunzip -c busybox-1.00-pre2.tar.gz | tar xvf -
```

5. Pindah ke direktori uClibc-0.9.13

```
debian:/myboot# cd uClibc-0.9.13
debian:/myboot/uClibc-0.9.13#
```

6. Buat simbolik link dengan perintah berikut:

261

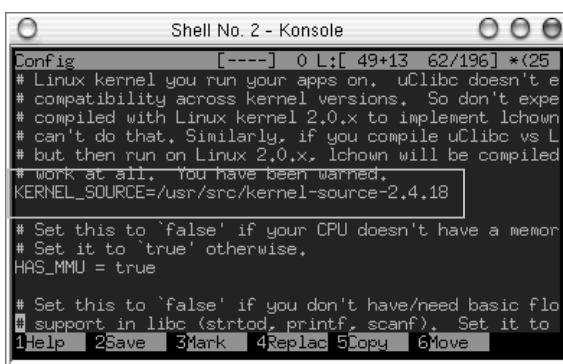
```
debian:/myboot/uClibc-0.9.13# ln -s  
\\ ./extra/Configs/Config.i386 ./Config
```

7. Edit file Config dengan menggunakan text editor (misal: mcedit, vi, dll)

```
debian: /myboot/uClibc-0.9.13# mcedit Config
```

Kemudian lakukan perubahan pada file-file berikut:

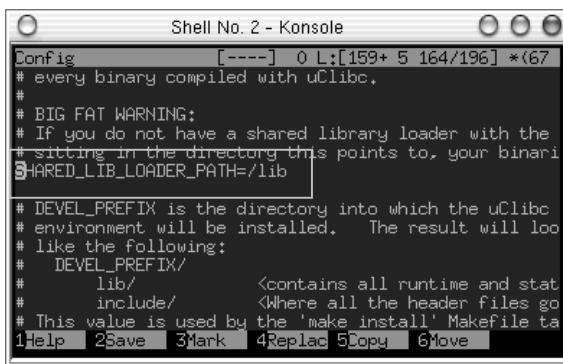
```
KERNEL_SOURCE=/usr/src/kernel-source-2.4.18 (letak source kernel)
```



```
Config [----] 0 L:[ 49+13 62/196 ] *(25  
# Linux kernel you run your apps on. uClibc doesn't e  
# compatibility across kernel versions. So don't expe  
# compiled with Linux kernel 2.0.x to implement lchown  
# can't do that. Similarly, if you compile uClibc vs L  
# but then run on Linux 2.0.x, lchown will be compiled  
# work at all. You have been warned.  
KERNEL_SOURCE=/usr/src/kernel-source-2.4.18  
  
# Set this to 'false' if your CPU doesn't have a memor  
# Set it to 'true' otherwise.  
HAS_MMU = true  
  
# Set this to 'false' if you don't have/need basic flo  
# support in libc (strtod, printf, scanf). Set it to  
1:help 2:Save 3:Mark 4:Replace 5:Copy 6:Move
```

Gambar 13.1 Kernel source

```
SHARED_LIB_LOADER_PATH=/Lib
```



```
Config [----] 0 L:[159+ 5 164/196 ] *(67  
# every binary compiled with uClibc.  
#  
# BIG FAT WARNING:  
# If you do not have a shared library loader with the  
# sitting in the directory this points to, your binari  
SHARED_LIB_LOADER_PATH=/lib  
  
# DEVEL_PREFIX is the directory into which the uClibc  
# environment will be installed. The result will loo  
# like the following:  
# DEVEL_PREFIX/  
#     lib/           <contains all runtime and stat  
#     include/        <Where all the header files go  
# This value is used by the 'make install' Makefile ta  
1:help 2:Save 3:Mark 4:Replace 5:Copy 6:Move
```

Gambar 13.2 Letak share library

```
DEVEL_PREFIX=/home/kari/myboot/uclibc-dev
```

```

Config      [----] 0 L:[167+ 8 175/196] *(72
# environment will be installed.  The result will loo
# like the following:
#  DEVEL_PREFIX/
#      lib/           <contains all runtime and stat
#      include/        <Where all the header files go
# This value is used by the 'make install' Makefile ta
# directory is compiled into the uclibc cross compiler
# have to recompile if you change this value...
DEVEL_PREFIX = /home/kari/myboot/uclibc-dev

# SYSTEM_DEVEL_PREFIX is the directory prefix used whe
# bin/arch-uclibc-gcc, bin/arch-uclibc-ld, etc.  This
# the 'make install' target, and is not compiled into
# defaults to $DEVEL_PREFIX/usr, but makers of .rpms a
1Help 2Save 3Mark 4Replace 5Copy 6Move

```

Gambar 13.3 Letak devel_prefix Uclibc

8. Ketikkan perintah berikut:

```

debian:/myboot/uClibc-0.9.13# make
debian:/myboot/uClibc-0.9.13# make install
debian:/myboot/uClibc-0.9.13# make      \
PREFIX=/home/kari/myboot/rootfs install_target

```

Jika terdapat pesan error diakhir eksekusi program, coba anda perhatikan pesan error tersebut, kemudian lakukan perubahan sesuai informasi yang diberikan.

9. Pindah ke sub-direktori busybox-1.00-pre2

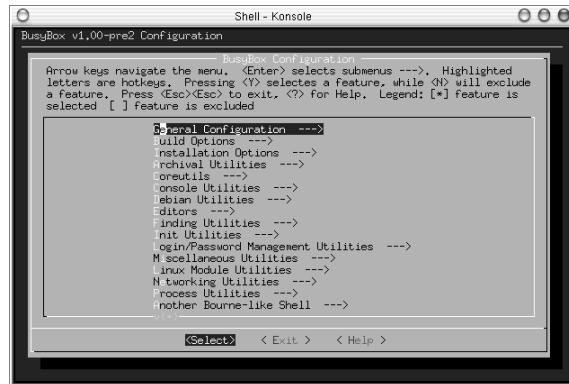
```

debian:/myboot/uClibc-0.9.13# cd ../busybox-1.00-pre2
debian:/myboot/busybox-1.00-pre2# ls

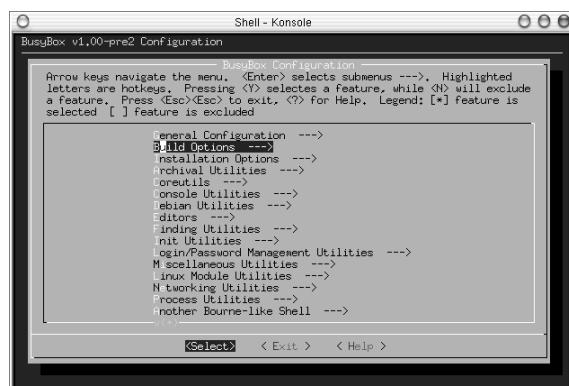
```

10. Pada tahap ini akan dilakukan konfigurasi busybox dengan mengetikkan perintah berikut:

```
debian:/myboot/busybox-1.00-pre2# make menuconfig
```

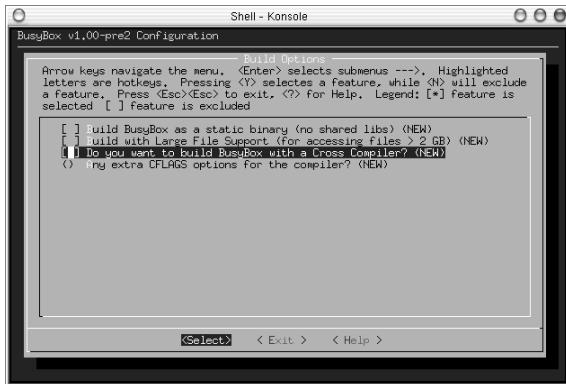


Gambar 13.4 Konfigurasi BusyBox

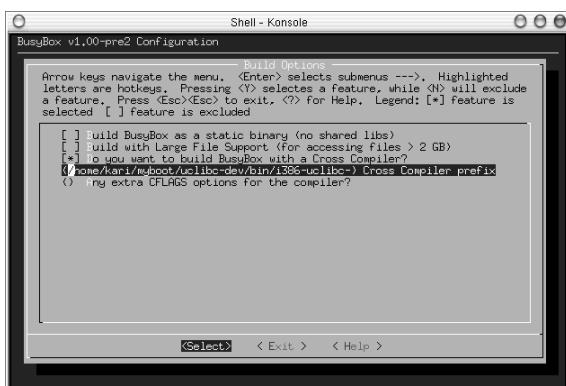


Gambar 13.5 Menu BusyBox

Pada menu [Build Options], pilih opsi “do you to build BusyBox with a cross compiler? (NEW)” kemudian edit letak prefix uclibc seperti yang tampak pada gambar 13.7



Gambar 13.6 BusyBox dengan cross compiler



Gambar 13.7 Letak prefix cross compiler

Tambahkan modul-modul yang akan digunakan sesuai kebutuhan anda dengan menekan tombol 'spacebar keyboard' pada opsi modul yang anda butuhkan. Kemudian ketikkan perintah berikut:

```
debian:/myboot/busybox-1.00-pre2# make dep && make && \ make  
PREFIX=/home/kari/myboot/rootfs install
```

Jika tidak terdapat pesan error, maka anda telah berhasil menginstal busybox.

11. Kemudian ketikkan perintah berikut untuk menjalankan program direktori tertentu karena direktori /lib belum terinstal pada sistem host anda.

```
debian:~# chroot /home/kari/myboot/rootfs /bin/sh  
(Ctrl + D) untuk keluar dari sub-shell chroot.
```

Tahap Instalasi

Seluruh librari dan file-file *executable* telah terinstal ke dalam direktori rootfs. Tahap selanjutnya adalah pembuatan direktori yang nantinya akan digunakan di floppy drive.

1. Pindah ke direktori rootfs kemudian buat direktori seperti contoh berikut:

```
debian:~# cd myboot/rootfs  
debian:/myboot/rootfs# mkdir -p dev tmp etc proc mnt \ etc/init.d
```

Kemudian tambahkan node-node device seperti floppy disk, terminal, ram disk.

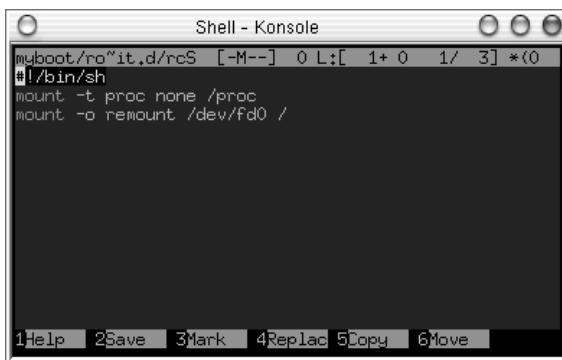
```
debian:/myboot/rootfs# mknod fd0 b 2 0  
debian:/myboot/rootfs# mknod tty c 5 0  
debian:/myboot/rootfs# mknod ttym c 4 1  
debian:/myboot/rootfs# mknod ram b 1 1  
debian:/myboot/rootfs# mknod mem c 1 1  
debian:/myboot/rootfs# mknod kmem c 1 2  
debian:/myboot/rootfs# mknod null c 1 3  
debian:/myboot/rootfs# mknod zero c 1 5
```

2. Salin direktori **init** dari source busybox-1.00.pre2 yang telah terinstal pada sistem anda.

```
debian:/myboot/rootfs# cd ../busybox-1.00.pre2  
debian:/myboot/busybox-1.00-pre2# ls  
debian:/myboot/busybox-1.0.0.pre2# cp -R init ..../rootfs/etc/
```

3. Kemudian buat script rcS di direktori etc/init.d/rcS

```
debian:/myboot/busybox-1.0.pre2# cd ../../rootfs/etc/init.d  
debian:/myboot/rootfs/etc/init.d# mcedit rcS
```



Gambar 13.8 Script executable

Kemudian buat file tersebut menjadi file *executable*

```
debian:/myboot/rootfs/etc/init.d# chmod +x rcS
```

4. Ubah kepemilikan seluruh file yang ada di rootfs oleh root

```
debian:~# chown -R 0:0 myboot/rootfs
```

Membangun Kernel

Berbeda dengan membangun kernel yang akan digunakan pada media hardisk, Kernel yang akan dibangun pada media floppy harus benar-benar dikustomasi sehingga ukurannya dapat dimuat pada media floppy drive.

Penulis menempatkan *source* kernel di direktori /usr/src/.

```
debian:~# cd /usr/src  
debian:/usr/src# ls  
kernel-source-2.4.18.deb
```

```
debian:/usr/src# dpkg -i kernel-source-2.4.18.deb
debian:/usr/src# ls
kernel-source-2.4.18.tar.bz2

debian:/usr/src# tar xjvf kernel-source-2.4.18.tar.bz2
atau
debian:/usr/src# bunzip2 -c kernel-source-2.4.18.tar.bz2 | tar xvf
-
debian:/usr/src# cd kernel-source-2.4.18
debian:/usr/src/kernel-source-2.4.18# make menuconfig
```

Berikut modul-modul yang akan diikutsertakan:

Modul-modul yang harus diikutsertakan ke dalam kernel:

Code maturity level->Prompt for development

Processor type and features->Support for your processor

General setup->kernel support for ELF binaries

Block devices->Normal PC floppy disk support

Block devices->RAM disk support

File systems->/proc filesystem

File systems->/dev file system support

File systems->Automatically mount at boot (option under devfs) File systems->Second extended fs

Character devices->Virtual terminal

Character devices->Support for console on virtual terminal

Console drivers->VGA text console

Modul-modul yang sifatnya optional:

General setup->PCI support

Networking options->Network packet filtering

Networking options->IP: Netfilter Configuration->Connection tracking

Networking options->IP: Netfilter Configuration->IP tables support

Networking options->IP: Netfilter Configuration->Full NAT

Networking options->IP: Netfilter Configuration->MASQUERADE

Network device support->Your Network device

Network device support->PPP

Network device support->PPP *

Character devices->Standard/generic

Character devices->Unix98 PTY

Sound->Sound card drivers (mp3 jukebox)

Modul yang dihilangkan:

General setup->PCI device name database

Setelah memilih semua modul yang akan diikutsertakan, ketikkan perintah berikut:

```
debian:/usr/src/kernel-source-2.4.18# make clean  
debian:/usr/src/kernel-source-2.4.18# make dep  
debian:/usr/src/kernel-source-2.4.18# make bzImage  
debian:/usr/src/kernel-source-2.4.18# make modules  
debian:/usr/src/kernel-source-2.4.18# make modules_install
```

Jika tidak terdapat pesan error pada akhir eksekusi program, maka anda telah berhasil menginstall kernel. Pada subdirektori 'arc/boot' akan terdapat file kernel image '**bzImage**'. Hasil kompilasi kernel yang penulis buat berukuran '372 kb'. Jika anda ingin memeriksa kapasitas kernel image yang anda kompilasi, gunakan perintah berikut:

269

```
debian: /usr/src/kernel-source-2.4.18# cd arc/i386/boot/  
debian: /usr/src/kernel-source-2.4.18/arc/i386/boot# du -k bzImage  
372  bzImage
```

Membuat disket bootable

1. Format disket anda dengan perintah berikut

```
debian:~# fdformat -n /dev/fd0
```

2. Buat filesystem ext2 pada disket yang akan digunakan. Perintah yang digunakan adalah:

```
debian:~# mke2fs /dev/fd0
```

3. Buat sub-direktori 'mnt' di direktori 'myboot'

```
debian:~# cd myboot/  
debian:/myboot# mkdir mnt
```

4. mount disket anda ke path myboot/mnt

```
debian:~# mount /dev/fd0 myboot/mnt
```

5. Salin seluruh isi direktori 'rootfs' ke dalam subdirektori 'mnt' yang telah dibuat.

```
debian:/myboot# cd rootfs/  
debian:/myboot/rootfs# cp -a * ../mnt  
debian:/myboot/rootfs# umount /dev/fd0
```

Lilo Boot loader

1. Mount kembali floppy drive anda dengan perintah berikut:

```
debian:~# mount /dev/fd0 myboot/mnt/
```

270

Debian GNU/Linux 2nd Edition

Askari Azikin

©2004-2007, <http://www.debianindonesia.org>

E-mail: kari@debianindonesia.org

2. Buat sub-direktori 'boot' di direktori '/mnt/

```
debian:~# mkdir myboot/mnt/boot
```

3. Salin file-file yang berada di direktori '/boot' anda ke direktori '/home/kari/myboot/mnt/boot'

```
debian:~# cp -a /boot/* myboot/mnt/boot
```

Setelah langkah di atas selesai, salin kernel image yang telah dikompilasi dengan perintah berikut:

```
debian:~# cp /usr/src/kernel-source-2.4.18/arc/i386/boot/bzImage  
/home/kari/myboot/mnt/boot/  
debian:/myboot/mnt/boot# ls  
bzImage  
boot.b  
map
```

4. Buat file 'lilo.conf' di direktori '/home/kari/myboot/'

```
debian:/myboot/mnt/boot# cd ../../  
debian:/myboot# mcedit lilo.conf
```

Kemudian tambahkan baris berikut:

```
boot=/dev/fd0  
install=/home/kari/myboot/mnt/boot/boot.b  
map=/home/kari/myboot/mnt/boot/map  
delay=50  
compact  
image=/home/kari/myboot/mnt/boot/bzImage  
initrd=/home/kari/myboot/mnt/boot/root.img.gz  
label=GNU/Linux
```

```
root=/dev/fd0
```

Kemudian ketikkan perintah berikut:

```
debian:~# lilo -C /home/kari/myboot/lilo.conf
debian:~# umount /dev/fd0
```

RAM disk

Salah satu keuntungan penggunaan RAM disk adalah load program akan lebih cepat.

```
debian:~# dd if=/dev/zero of=/homr/kari/myboot/root.img bs=1k
count=1000
```

```
debian:~# mke2fs -F -N 200 /home/kari/myboot/root.img
```

Kemudian mount image tersebut dengan menggunakan tambahan option '*-o loop*'.

```
debian:~# mount -o loop myboot/root.img myboot/mnt/
```

Tahap selanjutnya adalah mengkompress file 'root.img' dengan perintah berikut:

```
debian:~# gzip -9 myboot/root.img
```

Kemudian salin file tersebut ke direktori home/kari/myboot/mnt/boot/. Perintah yang digunakan adalah:

```
debian:~# cp myboot/root.img mnt/boot/
```

Reboot komputer anda dan ubah *first boot* dari floppy drive pada BIOS.