Programmation Système
© 2005 Brice Goglin
Brice.Goglin@ens-lyon.org

Compilation de modules noyau Linux

On supposera ici vouloir compiler un module nommé mymodule à partir de deux fichiers sources mymodule-core.c et mymodule-extra.c et d'un fichier d'en-tête mymodule.h.
Tout ce qui suit est à peu près documenté dans le sous-répertoire Documentation/kbuild/ des sources du noyau.

Compiler à la main

Jusqu'au noyau 2.4, il est possible d'utiliser un Makefile à peu près normal pour compiler un module noyau à la main.

```
KERNEL_DIR      =       /usr/src/linux-2.4.26
CC      =       gcc
LD      =       ld -r
CFLAGS  =       -Wall -D__KERNEL__ -DMODULE -I $(KERNEL_DIR)/include

MODULE  =       mymodule.o
MODULE_SRCS     =       mymodule-core.c mymodule-extra.c
MODULE_OBJS     =       $(subst .c,.o,$(MODULE_SRCS))

all: $(MODULE)

$(MODULE): $(MODULE_OBJS)
<tab> $(LD) $< -o $@

%.o: %.c
<tab> $(CC) $(CFLAGS) -c $< -o $@

clean:
<tab> rm -f *~ $(MODULE) $(MODULE_OBJS)

depend:
<tab> $(CC) $(CFLAGS) -MM $(MODULE_SRCS) > .depend

-include .depend
```

En fait, si votre module n'a qu'un seul fichier source, l'étape d'édition de lien (avec $(LD)) devient inutile.

Utiliser le Makefile du noyau

Depuis les noyaux 2.6, compiler à la main n'est plus possible facilement. Il faut utiliser le Makefile du noyau. Cependant, cette partie étant encore en développement, elle évolue régulièrement.
Ce qui suit est censé marcher depuis 2.6.6. Pour les noyaux plus vieux, la cible clean pourra ne pas fonctionner et il faudra remplacer M= par SUBDIRS=.

Dans le répertoire où se trouve nos sources à compiler, on utilise alors un makefile de la forme :

```
ifdef KERNELRELEASE
# partie utilisée par le Makefile du noyau
obj-m   :=      mymodule.o
mymodule-y      :=      mymodule-core.o mymodule-extra.o

else
# partie utilisée initialement, qui va appeler le Makefile du noyau
KERNEL_DIR = /usr/src/linux-2.6.10

all:
<tab> make -C $KERNEL_DIR M=`pwd`

clean:
<tab> make -C $KERNEL_DIR M=`pwd` clean

endif
```

Noter que la compilation va en fait produire mymodule.ko et non mymodule.o comme en 2.4.

Depuis les noyaux 2.6, il n'y a plus besoin de vérifier les dépendances. D'où l'absence de cible depend.

Depuis 2.6.10, il est en fait possible de séparer ce Makefile en deux fichiers. La partie entre ifdef et else, va dans un fichier nommé Kbuild. Celle entre else et endif reste seule dans le Makefile. Cela rend le tout beaucoup plus lisible.

Bref, pour compiler un module pour un noyau Linux 2.6, mieux vaut utiliser un noyau récent.

Utiliser le Makefile du noyau 2.4

Il est en fait possible d'utiliser un Makefile du type 2.6 avec un noyau 2.4. Cependant, il faudra écrire les cibles clean et depend à la main. De plus, il faut ajouter ce qui suit avant le else :

```
mymodule.o: $(mymodule-y)
<tab> $(LD) -r -o $@ $(mymodule-y)
```