Building a kernel module for many distribuitions

Rami Rosen
ramirose@gmail.com

Haifux – Lightning Talks , July 2008

Building a kernel module for many distribuitions

We want to build a kernel module for FC4,FC5,FC6,FC7,FC8,FC9 and EL4,EL5 and maybe more distribuitions.
If we will try to **insmod** a module built on FC4 to ,say, FC9, we will get an error:
insmod: error inserting 'myModule.ko': -1 Invalid module format
In syslog , you will see:
version magic xxx should be yyy.
xxx = module vermagic; yyy = kernel vermagic

Building a kernel module for many distribuitions

- modprobe -f or modprobe --force will not help in this case.

Building a kernel module for many distribuitions-contd

- Behind the scenes:
  - insmod/modprobe call a system call, *sys_init_module()* *(kernel/module.c.)*
  - -f (--force) options is the same as --force-vermagic and –force-modversion together; see man modprobe.
- See also: include/linux/vermagic.h and:
  - include/linux/version.h
  - For 2.6.25 it has:
  - #define LINUX_VERSION_CODE 132633
  - 132633 is 20619 => a hex representation of 2.6.25

Building a kernel module for many distribuitions-contd

- You can see the vermagic of a module by:
- modinfo -F vermagic myModule.ko
  - (-F stands for field)
  - 2.6.23.1-42.fc8 SMP mod_unload
  - (BTW: mod_unload says that the kernel was build with CONFIG_MODULE_UNLOAD set. Most distros enable module unloading; when this is not the case, running “rmmod myModule” will cause this:
  - FATAL: Kernel does not have unload support.
- Another way is by :
- **objdump myModule.ko --full-contents --section=.modinfo**

Building a kernel module for many distribuitions-contd

- You can also use **readelf --sections**

Building a kernel module for many distribuitions-contd

- Note:
- If the kernel version is 2.6.x.y.z, then modules

  built against 2.6.x.y will load into 2.6.x.y.* kernels.
- For example:
  - A module built against 2.6.25.9-76.fc9.x86_64 (latest FC9 kernel-devel) will load into 2.6.25 kernel

Building a kernel module for many distribuitions-contd

- Not always we have machines available on which we have all these environments.
- Even if we have, and we build on each of them

the module, we have the overhead of

synchronization between them.

Building a kernel module for many distribuitions-contd

- There are several solutions.
- For example, virtualization, chroot.
- Proposed solution:
- First, install the kernel-devel rpm of each distro.
  - This rpm include (mostly) kernel header files.
- Find out which gcc is for each distro and install the corresponding **source** rpm.
- For example:
- For FC6 we have gcc-4.1.1-30:
- Run: rpm -ivh gcc-4.1.1-30.src.rpm.

Building a kernel module for many distribuitions-contd

- Caveat: if you are building the module for fewer distros (two, three...) than sometimes you may have a suitable compat-gcc pacakge ready.
- Then: rpm -bp /usr/src/redhat/SPECS/gcc41.spec
- The gcc source code is somewhere under /usr/src/redhat/BUILD/
- Then, you should build gcc.
- It is important to create a folder for the build.

mkdir build; cd build

Building a kernel module for many distribuitions-contd

- ../configure --prefix=/work/tools/fc6/gcc --enable-threads=posix –enable-languages=c
- Then make and make install.
- What is inaccurate here ?
  - We better use the configuration options with which the gcc from the selected distro was built. We can
  - Get these configuration options in the spec file of the gcc
  - To be on the safe side, it is also better to build the gcc against the corresponding binutils package.

Building a kernel module for many distribuitions-contd

- Create a folder (let's say myFolder) which will have your module source code and header files. Let's say they are myModule.c and myModule.h.
- Create subfolders, on for each distro:
  - myFolder/fc4,myFolder/fc5, etc.

Building a kernel module for many distribuitions-contd

- Create soft links for myModule.c,myModule.h in each subfolder.
- Then, create for each distro a Makefile in its corresponding folder.
- Example for a Makefile for FC6:

```
obj-m    := myModule.o

CC:= /work/tools/fc6/gcc/bin/gcc

KDIR:=/usr/src/kernels/2.6.18-1.2798.fc6-x86_64

PWD    := $(shell pwd)


default:

$(MAKE) CC=$(CC)  -C $(KDIR) SUBDIRS=$(PWD) modules
```

Building a kernel module for many distribuitions-contd

- In case you need to add #ifdef in the code for different kernel, there are two ways to do it:
  - EXTRA_CFLAGS in the makefile
    - For example:
    - EXTRA_CFLAGS := -DKER2625
    - And than , in the code, #ifdef KER2625
  - Adding using LINUX_VERSION_CODE /KERNEL_VERSION from include/linux/version.h
  - #if LINUX_VERSION_CODE >= KERNEL_VERSION(2,2,0)
  - ....
  - #endif

Building a kernel module for many distribuitions-contd

- In myFolder, create a makefile which will call make on all subfolders:
- For example:

```
#Makefile

all:

cd fc4 && $(MAKE)

cd fc5 && $(MAKE)

cd fc6 && $(MAKE)

....

cd el5 && $(MAKE)
```

Building a kernel module for many distribuitions-contd

- **Thank you!**