



Author: Jon Barnett
Date: 17th October 2003

Slackware 9.1: Rebuilding the kernel

I have a Dell Inspiron 8100 laptop. It has a few quirks in that there is a PCMCIA wireless LAN card, an NVIDIA graphics card, a Fast InfraRed (FIR) port, a Maestro3 sound system and an inbuilt Ethernet-Modem combination. Slackware 9.1 does an admirable job of getting the system running. There are a few areas that may need some manual help. I'll outline these so that it gives an idea of things that might get broken later.

The PCMCIA manager does not work well in detecting allocated resources. The first problem was with booting Slackware 9.1 after completing the installation from the CD distribution. For the Inspiron 8100, the boot process stalled when checking the PCMCIA resources; the memory to be exact.

I rebooted with the Slackware distribution CD 1 and made the following change to `/etc/pcmcia/config.opts`:

```
include port 0x100-0x4ff, port 0x800-0x8ff, port 0xc00-0xcff
```

to:

```
include port 0x100-0x4ff, port 0xc00-0xcff
```

This worked sufficiently well enough that the boot completed and the wireless card, a Cabletron detected as a Lucent Agere started and connected to the network. I had to manually add the second network interface to `/etc/rc.d/rc.inet1.conf` and the 3COM WinModem was unusable as no drivers exist for this particular manufacturer.

The PCMCIA module allocated the same IRQ (3) as `ttyS3`, which was configured for the FIR. The FIR was not detected either so the `smc-ircc` driver was also not loaded. In order to force the PCMCIA manager to not allocate the reserved IRQ for its devices, I added the following exclusion clause to `/etc/pcmcia/config.opts`:

```
# Second built-in serial port  
exclude irq 3
```

So remember to check all your port and IRQ allocations after you have booted with PCMCIA and adjust things accordingly so that you don't get any critical resources shared. This will avoid otherwise inexplicable lock-ups when running your system. I went back to the BIOS and checked the serial port was allocated to the FIR.

Having performed this and rebooted to ensure that the changes worked, I found the FIR was still not enabled. The first thing I noticed was that Slackware 9.1 didn't seem to have the IrDA utilities. I downloaded and built the utilities from the [Linux-IrDA Project](#). I read through all the instructions and by trial and error concluded the best configuration consisted of forcing the IrDA module configuration by adding an `/etc/rc.d/rc.irda` script to the boot process.

In /etc/modules.conf I added the following:

```
alias tty-ldisc-11 irtty
alias char-major-161 ircomm-tty
alias irda0 smc-ircc
```

The script to start and stop IrDA was:

```
#!/bin/sh
# Start/stop/restart IrDA.
# Start IrDA:
irda_start() {
    echo "Starting IrDA"
    /sbin/setserial /dev/ttyS3 uart none
    /sbin/insmod irda
    /sbin/insmod irport
    /usr/sbin/irattach irda0 -s
}
# Stop IrDA:
irda_stop() {
    killall irattach
}
# Restart IrDA:
irda_restart() {
    irda_stop
    sleep 1
    irda_start
}
case "$1" in
'start')
    irda_start
    ;;
'stop')
    irda_stop
    ;;
'restart')
    irda_restart
    ;;
*)
    echo "usage $0 start|stop|restart"
esac
```

I added the IrDA startup in the `/etc/rc.d/rc.M` script, inserting it just after the configuration of the basic network service.

```
if [ -x /etc/rc.d/rc.inet1 ]; then
    . /etc/rc.d/rc.inet1
fi
```

```
# Start IrDA:
```

```
if [ -x /etc/rc.d/rc.irda ]; then
    . /etc/rc.d/rc.irda start
fi
```

You can similarly stop the service by adding the appropriate script test and execute code to `/etc/rc.d/rc.6`.

I created the necessary devices as specified by the Linux Infrared HOWTO with one exception. I created an `irda` group, restricted the permissions so that only the `irda` group could read and write to the IrDA device, and put only approved users in the group. This gives you better security control.

```
mknod /dev/ircomm0 c 161 0
mknod /dev/ircomm1 c 161 1
mknod /dev/irlpt0 c 161 16
mknod /dev/irlpt1 c 161 17
mknod /dev/irnet c 10 187
chmod 660 /dev/ir*
chown root:irda /dev/ir*
usermod -G irda useraccount
```

I checked the BIOS configuration to make sure the laptop was set for FIR and on `ttyS3`, and rebooted to Linux but `irattach` would not stay running so there was no IrDA interface when you ran `ifconfig`. There was a post that alluded to a Flash BIOS problem. Reports indicated that revision A10 would fix the problem. I had revision A10 and it didn't work. So I tried upgrading to the latest which was A15 and the IrDA port started working after another reboot. I also performed a cold reboot at this stage.

Checking the status by `ifconfig` returned the following:

```
irda0      Link encap:IrLAP  HWaddr 61:df:b6:52
           UP RUNNING NOARP  MTU:2048  Metric:1
           RX packets:8 errors:0 dropped:0 overruns:0 frame:0
           TX packets:4010 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:8
           RX bytes:248 (248.0 b)  TX bytes:127104 (124.1 Kb)
           Interrupt:3 Base address:0x2e8
```

The modules and module dependencies shown by `lsmod` were:

```
smc-ircc          7520    1
irport           5320    1 [smc-ircc]
irda             94480    0 [smc-ircc irport]
```

Finally, testing the detection of a device by placing my Siemens mobile (cell) phone in front of the IrDA port on the right side of the laptop and checking the contents of `/proc/net/irda/discovery` resulted in this.

IrLMP: Discovery log:

```
nickname: SIEMENS ME45, hint: 0xb124, saddr: 0x61dfb652, daddr:
0x08922407
```

The system was now operational and ready to be upgraded with Dropline Gnome.

Dropline Gnome 2.4

Dropline Gnome 2.4 is an i686 optimized Gnome distribution for Slackware. It has recently been released and patches well into Slackware 9.1. The installation went without a problem although you need to be online to first download the subsystems or alternately, order and install from CDs.

Configuring Gnome for a wheel mouse means that you can take advantage of the wheel for every application packaged with Dropline Gnome. This was pretty impressive. You also do not need to install a special mouse driver external to Gnome.

Recompiling the kernel

Now that you have an i686-optimized Gnome system, you probably should recompile your kernel for the i686 architecture rather than the vanilla i486 supplied with Slackware 9.1.

The kernel is straightforward to recompile if you install the Linux source supplied with the Slackware 9.1 distribution. It has already been configured for the settings used to build the base kernel supplied with Slackware 9.1. In the simplest case, you can change directory to `/usr/src/linux`, just “make menuconfig”, change the “Processor type” to Pentium III, quit and confirm that you want to save the new configuration. For the more adventurous, you can also change other options such as removing support for older devices, network cards you don't use and so on. This will shrink the kernel size a bit. When you are ready to build and install the new kernel, just run the following:

```
make dep; make bzImage; make modules; make modules_install
```

This assumes that everything is going to build successfully and install. You may want to take it one step at a time if you are more conservative. Remember to check for a string like “-march=i686” during the build. It will indicate that you are indeed building an i686 kernel.

At the end of this, you will need to manually install the System.map and kernel image. The following shows my approach to this, and this preserves the existing kernel just in case.

```
cp /usr/src/linux/arch/i386/boot/bzImage /boot/vmlinuz-ide-2.4.22-i686
cp /usr/src/linux/System.map /boot/System.map-ide-2.4.22-i686
ln -sf /boot/System.map-ide-2.4.22-i686 /boot/system.map
ln -sf /boot/vmlinuz-ide-2.4.22-i686 /boot/vmlinuz
lilo
```

This installs your new kernel and notifies the lilo boot loader of the presence of the kernel to be used. You may now reboot to see if your new kernel and main modules work. The original Slackware 9.1 kernel and symbols map, System.map-ide-2.4.22 and vmlinuz-ide-2.4.22 remain intact.

NVIDIA graphics card

By rebuilding your kernel and modules and installing these changes, you will lose your external NVIDIA graphics driver (module). So you will need to re-install this after performing your kernel and main modules installations. This should be a straightforward task, following the NVIDIA interactive run script. I ran the current driver at the time of writing as follows:

```
sh NVIDIA-Linux-x86-1.0-4496-pkg2.run
```

You will need to be on-line so the script can obtain source code to perform the driver module recompilation.

Should you not have run the NVIDIA installer previously, be aware that it will also modify /etc/modules.conf by adding the following line:

```
alias char-major-195 nvidia
```

Other than that, there is little else involved in the installation of the driver. You will need to modify your /etc/X11/XF86Config or /etc/X11/XF86Config-4 files, depending on your choice. Note that the XF86Config-4 file overrides the XF86Config file if they both exist. The NVIDIA readme.txt explains this in more detail. I copied the file supplied with the NVIDIA distribution rather than trying to follow the NVIDIA instructions for modifying the configuration files supplied with Slackware 9.1. I copied the fonts section from the existing XF86Config-4 file into the NVIDIA file. The sample configuration file, the readme.txt file, the C header files and further information on your NVIDIA driver can be found in /usr/share/docs/NVIDIA_GLX-1.0.

I also copied my wheel mouse configuration into the new configuration file. After that it was simply a matter of tweaking the video selection in the configuration to take advantage of 1400x1050 resolution.

All being well, you should see an NVIDIA splash screen appear before entering Gnome.

ALSA sound configurations

Rebuilding the kernel and re-installing the modules also destroys your existing ALSA sound modules for the kernel. This is because ALSA is not currently bundled into the kernel build process. ALSA is touted to be bundled into the 2.6 release kernel but I cannot confirm that at this time.

In the meantime, you will need to obtain the ALSA driver package from the [ALSA Project](#) website. You won't need the other packages as Slackware and Dropline Gnome supply the tools, utilities and libraries. In the case of Dropline Gnome, these additional programs and libraries are going to be optimized for the i686 architectures anyway.

The instructions for the driver compilation are straight forward and also provided on the ALSA Project website.

Assuming that your Linux source is installed at `/usr/src/linux`, create a directory for your ALSA driver package at `/usr/src/alsa`. Change to that directory and unpack your package into that directory. Your drivers will be extracted into a subdirectory under `/usr/src/alsa`. For the current 0.9.7c driver package, the full directory path is `/usr/src/alsa/alsa-driver-0.9.7c`. Use the standard `./configure`, `make`, `make install` to build and install your new ALSA drivers.

The ALSA driver build configuration requires the Linux source to be already configured and available at `/usr/src/linux`. You can override this with the `-with-kernel=/usr/local/src/linux` or whatever is appropriate for the location of your Linux source. However, for a standard Slackware system, the default is fine. My actual configuration for an i686 build was:

```
CFLAGS=-DCPU=686 ./configure
```

```
make
```

```
make install
```

You can reboot your system but you will probably discover like me, that the system has a few problems. In my case, the kernel could not find various sound-related modules and there were some problems with the IrDA. The Inspiron 8100 has an SMC-based IrDA port and I have the `smc-ircc` module loaded for FIR. The original Slackware ALSA drivers ensured that everything ran without problems and the Maestro3 sound system did not clash with the IrDA. For some reason, after the rebuild the kernel reported the following in `/var/log/messages`:

```
Oct 13 15:02:56 drak kernel: NETDEV WATCHDOG: irda0: transmit timed out
```

There were also a lot of messages regarding the sound system modules not being loaded. I tried to manually load the sound modules using `insmod` and `modprobe` with little success, and also tried changing the various obvious configuration files such as `/etc/modules.conf`. Rebooting resulted in the same errors. Out of frustration, I re-ran `alsaconf` and rebooted. This seemed to fix my boot problems. So in order to fix the final loading problems:

```
alsaconf
```

For those upgrading from Slackware 9.0 to 9.1, you might find that `alsaconf` is not installed as part of the upgrade. You can find this in the `utils` subdirectory of the ALSA driver package.

Running `alsaconf` might cause a few other problems with ownerships and permissions for devices if the installation believes this to be a new install. Normally, you create an audio group and make all sound related devices belong to that group. You make each user that you want to give access to the sound devices a member of the audio group by:

```
usermod -G audio useraccount
```

I had done this with the original Slackware 9.1 installation. When I ran `alsaconf` again after installing the new kernel, `alsaconf` seemed to overwrite my original working device ownerships. The devices reverted to the `sys` group.

In order to fix these:

```
chown root:audio /dev/snd/*
chown root:audio /dev/audio*
chown root:audio /dev/dm*
chown root:audio /dev/dsp*
chown root:audio /dev/midi*
chown root:audio /dev/mixer*
chown root:audio /dev/music*
chown root:audio /dev/pcmixer
```

This is actually a complete list of the sound devices but not all are used by the ALSA drivers. You can probably get by with fixing the ownership for just the `/dev/snd/*`, `/dev/dsp*` and `/dev/mixer*` devices. The driver package also has a script, `snddevices` that will properly set up your device permissions.

As a separate issue, occasionally while using some of the Dropline sound applications, I get messages like this in `/var/log/syslog`.

```
Oct 13 19:28:00 drak modprobe: modprobe: Can't locate module sound-
slot-1
```

```
Oct 13 19:28:00 drak modprobe: modprobe: Can't locate module sound-
service-1-0
```

You can also fix these by adding some lines to your `/etc/modules.conf`. An example for this would be:

```
alias sound-slot-1 off
alias sound-service-1-0 off
```

Do not add these within the `ALSACONF` sections as these lines will be overwritten next time you run `alsaconf`. Also, should you be adding new lines to `/etc/modules.conf` you should add them before the `ALSACONF` section. The reason for this is that next time you run `alsaconf` it will remove the old section and add the new `ALSACONF` to the end of the file. While this will not delete any additions you make to the file outside of the section, it could make your configuration start to look untidy in spacing and grouping of lines. This is purely an aesthetic consideration.

Final considerations

Any other drivers external to your standard kernel drivers will need to be similarly rebuilt after the kernel and main modules are put in place. These other modules are beyond the scope of this article to cover. However, this should give you a starting point for building a new kernel for your Slackware system, including getting the ALSA sound system running again.

Having made all these changes and coaxing the system back to an error-free loading state, you should be ready to run and use your upgraded Slackware 9.1/Dropline Gnome 2.4 installation.