Building a new Kernel

Philosophy: kernel contains device drivers, network code, program loader, file system code.
Load everything, kernel is large but most code is unused

Task: build the kernel you need.
Example: match an embedded system

Philosophy: Kernel bugs will be fixed.

Rebuild to (1) patch a bug, (2) install new capabilities, (3) streamline for efficiency

Rebuilding the kernel:
Linux: recompile the source code.

Warning: if the kernel doesn't include the hardware, it will not boot.

Principle: maintain a backup boot floppy or partition
You have to something that boots to restore the old kernel.

Modules: most capabilities are available in loadable modules. The kernel must be able to boot (correct disk or network drivers); other capabilities can be outside the core of the kernel.

Sources

Freely available.

Usually in `/usr/src/linux`
mostly this is a link
(e.g. to `/usr/src/linux.2.6.34`)
switching the link, switches your kernel version.

Two additional links are necessary, our release has them, but if you are using another release check to make sure these are there before you rebuild.

`ln -s /usr/src/linux/include/linux /usr/include/linux`
(the second is a link to the first).
The correct include files for your version of the kernel must be in the compilers include path.

`ln -s /usr/src/linux/include/asm /usr/include/asm`
The correct assembler for your machine kernel must be in the compilers include path.

linux has versions for the following CPUs x86 (PC), 68000 (Amiga), 604 (PowerPC/Mac), Sparc (Sun) , MIPS (SGI), Alpha (Compaq), Precision Architecture (HP), AMD64, ARM9

Configuring the Kernel

`cd /usr/src/linux`

Note: down to the `cp` you do not have to be root,
you just have to own `/usr/src/linux`

`make config` – you will be asked 20 questions about your system.
If you reach one you don't know,
record it and keep going.
You can repeat this command.
This command sets up the `Makefile` to build your version of the kernel.

`make menuconfig` – menu driven version
Allows you to make and unmake selections.

When you exit both save your selections as in a file called `.config` They also build a `Makefile`

It will use an existing `.config` or a default config.
Consequence: default answers reflect the previous config

`make dep` – make dependencies. Make should recompile files if they have changed or any .h file they include has changed. This command builds the lists of depencenies. Only needs to be done once when you add includes to kernel files.

Compiling the Kernel

`make bzImage` – (or `Image` or `vmlinuz` or `zImage`)
Compile a kernel, a `z` indicates a compressed version.
check your makefile to see what this is called.
Options for make are followed by colons
`vmlinuz: ...`

This builds your kernel and stores it in
`/usr/src/linux/arch/x86/boot`
For example a bzImage kernel would be found as
`/usr/src/linux/arch/x86/boot/bzImage`

Move the kernel into the boot area calling it
/boot/bzImage

To test your kernel you need to modify your boot configuration to support a dual boot and rebuild the boot record

Linux Loader

A brief review of LILO

Until the kernel is loaded,
the computer does know about directory structure
so you can't load a file.
LILO must work by knowing the block numbers of each kernel block to load.
New kernel means different block numbers.

The LILO setup program (also called lilo)
is used with Linux running,
it knows about directories.
It finds the kernel on disk and builds the master boot record with the correct block entries for that kernel.

Consequence: run `lilo` each time you install (mv or cp) a kernel

Lilo flexibility: you are allowed multiple sections, you can have multiple kernels...as long as they have different names.

LILO

`/sbin/lilo` – the program that installs lilo into the Master Boot Record (MBR)

`/etc/lilo.conf` – configuration file.
Tells `/sbin/lilo` what to put into MBR.
in effect tells lilo boot program how to behave;

`/sbin/liloconfig` – shell script that
helps you build `lilo.conf`
then invokes `/sbin/lilo` to install the lilo boot program

The boot sequence can be set:
1) boot directly, no options
2) prompt for boot option
3) delay (to allow left shift key to be pressed leisurely), then boot default option.

Sample `lilo.conf`

```
delay = 50
vga = normal
ramdisk = 0
lba32
# partition config
image = /boot/vmlinuz
  root = /dev/sda2
  label = linux
  read-only
# another partition config
other = /dev/sda3  # dos partition
  label = DOS
```

`delay` – delay in 10ths of second, then default
default is first partition in config
`prompt` – prompt and wait (not shown)
`ramdisk` – size of ramdisk (0 = no ramdisk)
`label` – what to type if we want this option
`loader` – program used to load OS found in this partition
`root` – linux root partition
`read-only` – initial mount is read-only for disk check

`image` usually a file (use the map), but can be a range of sectors. The kernel is assumed to be loaded in order into those sectors. (allows tape or floppy boot)

Lilo can be invoked directly:

`lilo -r /mnt -m /boot/map -C /etc/lilo.conf`
Non-standard files may be specified when lilo is invoked
defaults: `/boot/map` map of kernel `/etc/lilo.conf` configuration file.

lilo takes a large number of parameters (see Map Installer)
most override the configuration file.

Example: naming a boot device

`-b /dev/fd0` on command line

`boot=/dev/fd0` in lilo.conf

`-r /mnt` there is a disk mounted on `/mnt`. switch to that file system and run lilo using the `lilo.conf` found on that disk.

Boot Parameters

During the boot sequence

1) if the left shift key is depressed, a prompt is displayed for which partition to boot, then a prompt is displayed for boot parameters.

2) if caps lock is on a prompt is displayed for boot parameters. These override the config.

3) if there is a time-out, the default operating system (first partition listed) is booted

Parameters:

`no387` – disable FPU

`root=/dev/fd1` – designate root partition

`single` – single user mode, for maintenance

`ether=9,0x240,0,0,eth0` – network card has a non-standard configuration, this is where you should probe.
In `lilo.conf` the above parameter is entered as
`append="ether=9,0x240,0,0,eth0"`

Booting From Floppy

Method 1: entire system is on floppy

Often: boot floppy, root floppy

Often: root file system is loaded into RAM (ramdisk)

Method 2: kernel is on floppy, root is on hard drive

Method 3: lilo is on floppy, kernel is on hard drive

Only useful if you goofed MBR lilo install.

Examining Boot Messages

`dmesg`: a linux program that will display the later boot messages

You have to have the kernel loaded (decompressed) to save anything.

Booting Over A Network

There must be a boot server on the network.

The boot server will run:

`dhcpd` to supply a network identification
(`dhcpd` can also specify a kernel

`tftpd` to transfer the boot files to your machine

Your ROM BIOS must support network booting
(bootp, tftp and dhcp protocols)
or you may use a boot floppy/CD that has a netboot on it.

If you get into trouble on the kernel and can't boot, boot from the network (see prompt at bottom of boot screen);
mount your `/dev/sda2` on `/mnt`
adjust your `/mnt/etc/lilo.conf`
run the following series of commands then reboot:

```
mount -o bind /proc /mnt/proc
mount -o bind /sys /mnt/sys
chroot /mnt lilo
```