Kernel Seminar

Jalal Hajigholamali

Hadi Kianersi
Najibeh Pardis
Sepahrad Salour

2014 - 28 – April
Monday

What is LINUX?

Linux is a clone of the operating system Unix, written from scratch by Linus Torvalds with assistance from a loosely-knit team of hackers across the Net.

It aims towards POSIX and Single UNIX Specification compliance.

Kernel

It's a program that runs in Kernel Mode CPUs run either in Kernel Mode or in User Mode.

When in User Mode, some parts of RAM can't be addressed, some instructions can't be executed, and I/O ports can't be accessed.

When in Kernel Mode, no restriction is put on the program!

Besides running in Kernel Mode, kernels have three other peculiarities:

1. Large size (millions of machine language instructions)

2. Machine dependency (some parts of the kernel *must* be coded in Assembly language)

3. Loading into RAM at boot time in a rather primitive way

Kernel Entry Points

1. Software Interrupt
2. I/O Device Requires Attention
3. Time Interval Elapsed
4. Hardware Failure
5. Faulty Instruction

Why Is a Kernel So Complex ?

Large program with many entry points:

1. Must offer disk caching to lower average disk access time

2. Must support run nested kernel invocations

3. Must run with the interrupts enabled most of the time

4. Must be updated quite frequently to support new hardware circuits and devices

On What Hardware does it Run?

Although originally developed first for 32-bit x86-based PCs (386 or higher), today Linux also runs on (at least) the Compaq Alpha AXP, Sun SPARC and UltraSPARC, Motorola 68000, PowerPC, PowerPC64, ARM, Hitachi SuperH, Cell, IBM S/390, MIPS, HP PA-RISC, Intel IA-64, DEC VAX, AMD x86-64, AXIS CRIS, Xtensa, Tilera TILE, AVR32 and Renesas M32R architectures.

Linux is Portable

Linux is easily portable to most general-purpose 32- or 64-bit architectures as long as they have a paged memory management unit (PMMU) and a port of the GNU C compiler (gcc) (part of The GNU Compiler Collection, GCC).

Linux has also been ported to a number of architectures without a PMMU, although.

Linux Documanation

There is a lot of documentation available both in electronic form on the Internet and in books, both Linux-specific and pertaining to general UNIX questions.

I'd recommend looking into the documentation subdirectories on any Linux FTP site for the LDP (Linux Documentation Project) books.

README

There are various README files in the Documentation/subdirectory these typically contain kernel-specific installation notes for some drivers for example.

See Documentation/00-INDEX for a list of what is contained in each file.

Please read the Changes file, as it contains information about the problems, which may result by upgrading your kernel.

Installing

gzip -cd linux-3.X.tar.gz | tar xvf -

bzip2 -dc linux-3.X.tar.bz2 | tar xvf -

xz -dc linux-3.X.tar.xz | tar xvf -

Replace "X" with the version number of the latest kernel.

Patch

You can also upgrade between 3.x releases by patching.

Patches are distributed in the traditional gzip and the newer bzip2 format.

To install by patching, get all the newer patch files, enter the top level directory of the kernel source (linux-3.X) and execute.

```
gzip -cd ../patch-3.x.gz | patch -p1
bzip2 -dc ../patch-3.x.bz2 | patch -p1
xz -cd ../patch-3.x.xz | patch -p1
```

Replace "x" for all versions bigger than the version "X" of your current source tree,

Software Requirements

Compiling and running the 3.x kernels requires
- **Up-to-date versions of various software packages.**

Consult Documentation/Changes for the minimum version numbers required and how to get updates for these packages.

Configuring the kernel

Configuration Commands are:

1. make config
Plain text interface.

2. make menuconfig
Text based color menus, Radiolists & Dialogs.

3. make nconfig
Enhanced text based color menus.

4. make xconfig

X windows (Qt) based configuration tool.

5. make gconfig

X windows (Gtk) based configuration tool.

And more ...

Compiling the kernel

Make sure you have at least gcc 3.2 available.

For more information, refer to Documentation/Changes.

Do a "make" to create a compressed kernel image.

To do the actual install, you have to be root, but none of the normal build should require that.

Don't take the name of root in vain.

If you configured any of the parts of the kernel as 'modules',

You will also have to do "make modules_install".

Verbose kernel compile/build output

Normally, the kernel build system runs in a fairly quiet mode. (but not totally silent).

For this, use "verbose" build mode.

make V=1 all

Keep a backup kernel handy in case something goes wrong.

This is especially true for the development releases, since each new release contains new code which has not been debugged.

Backup /lib/modules/X

X is kernel version

e.g. 2.6.32.61

Boot with New Kernel

In order to boot your new kernel, you'll need to copy the kernel image

e.g.

.../linux/arch/i386/boot/bzImage

after compilation to the place where your regular bootable kernel is found. Normally /boot .

Something Goes Wrong

If you have problems that seem to be due to kernel bugs, please check the file MAINTAINERS to see if there is a particular person associated with the part of the kernel that you are having trouble with.

If there isn't anyone listed there, then the second best thing is to mail them to me:

torvalds@linux-foundation.org

and possibly to any other relevant mailing-list or to the newsgroup.

Bug Report

In all bug-reports, *please* tell what kernel you are talking about, how to duplicate the problem, and what your setup is (use your common sense)

Please read the REPORTING-BUGS document for details.

Thanks :)

Question ?