Building Customized Linux Kernels
A live demonstration
Mark Post
August 17, 2004
Session # 9280
Documentation

- The Linux Documentation Project
  http://www.tldp.org/

- Look for the “Kernel HOWTO”
  http://www.tldp.org/HOWTO/Kernel-HOWTO/
  http://www.digitalhermit.com/~kwan/kernel.html

- Practical experience here this week:
  Linux for S/390 Installation Lab, Tue. & Thu.
  9227 and 9230
Basic Process

- Get the source
- Unpack/install the source
- Apply IBM patches (if not already there)
- Generate a kernel configuration
  - make menuconfig
  - make oldconfig
  - make xconfig
  - make config
- Run
  - make dep
  - make image
  - make modules
  - make modules_install
Basic Process (2)

• Put new kernel into place
• Possibly regenerate the initrd
• Possibly update /etc/zipl.conf
• Run zipl
• Take the system down
• Boot from the new kernel
• Back off to the old kernel if necessary
Where to get the source

• “Pristine” source:

• Linux distribution-specific source:
  Usually included in your distribution installation media, or...
  https://portal.suse.com/

• IBM patches:
Unpack/Install the Source

• If you get a kernel source RPM, then install the source:
rpm -ivh kernel-source.rpm
  • Usually puts the source in /usr/src/linux-$VERSION

• If you downloaded source from ftp.kernel.org:
tar -zxvf linux-2.6.7.tar.gz
tar -jxvf linux-2.6.7.tar.bz2

• Don’t confuse this with a kernel SRPM
  • kernel-source-2.4.20.SuSE-62.i586.rpm
  • kernel-source-2.4.20.SuSE-62.src.rpm
  • kernel-source-2.4.20-8.i386.rpm
  • kernel-2.4.20-8.src.rpm
Unpack/Install the Source

• So what is the difference?
  • SRPM = vanilla source, patches, RPM spec file
gets installed into /usr/src/rpm/SOURCES
  gets processed with “rpmbuild -bb” command
  • RPM = updated source
gets installed into /usr/src/linux-$VERSION (usually)
   /usr/src/linux-2.4.19
Apply IBM patches

- Patches come in .tar.gz files.
- Contain a
  - LICENSE file (GPL)
  - .readme file
  - .diff file
- Read the .readme file(s) for patching order.
- cd to top-level directory and use patch command:
  `cat /path/to/diff.file | patch -p1 [ ---dry-run ]`
- Repeat for each .diff file.
Generate a kernel configuration

• Many ways to specify a particular kernel configuration:
  • make config (don't do this)
  • make oldconfig (used to start from a known configuration)
  • make menuconfig
  • make xconfig (don't do this on Linux/390)
Make menuconfig

Linux Kernel v2.4.19 Configuration

Main Menu

Arrow keys navigate the menu. <Enter> selects submenus. Highlighted letters are hotkeys. Pressing CY includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <??> for Help. Legend: [*] built-in [ ] excluded <M> module < > module capable

Code maturity level options --->
Loadable module support --->
Processor type and features --->
General setup --->
SCSI support --->
Block device drivers --->
Multi-device support (RAID and LVM) --->
Character device drivers --->
Network device drivers --->
Miscellaneous --->
Networking options --->
File systems --->
Kernel hacking --->

Load an Alternate Configuration File
Save Configuration to an Alternate File

[Select] < Exit > < Help >
Make menuconfig (2)
Don’t do this at home
<table>
<thead>
<tr>
<th>Feature</th>
<th>y</th>
<th>m</th>
<th>n</th>
<th>Help</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loopback device support</td>
<td>y</td>
<td>m</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Network block device support</td>
<td>y</td>
<td>m</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>RAM disk support</td>
<td>y</td>
<td>m</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Default RAM disk size</td>
<td>24576</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Initial RAM disk (initrd) support</td>
<td>y</td>
<td></td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>XPRAM disk support</td>
<td>y</td>
<td>m</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>S/390 block device drivers</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support for DASD devices</td>
<td>y</td>
<td>m</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Support for ECKD Disks</td>
<td>y</td>
<td>m</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Automatic activation of ECKD module</td>
<td>y</td>
<td></td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Support for FBA Disks</td>
<td>y</td>
<td>m</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Automatic activation of FBA module</td>
<td>y</td>
<td></td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Support for DIAG access to CMS reserved Disks</td>
<td>y</td>
<td>m</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>Automatic activation of DIAG module</td>
<td>y</td>
<td></td>
<td>n</td>
<td></td>
</tr>
</tbody>
</table>

Main Menu  Next  Prev
Usual order of commands:

- Save configuration file
- make mrproper (this wipes out .config!)
- copy saved configuration file to .config
- make menuconfig (or oldconfig)
- make dep (no longer needed in 2.6.x kernels)
- make image (on Intel, will likely be bzImage)
- make install (make sure you know what this does)
- make modules
- make modules_install
- depmod -a version-of-kernel-just-built
  - depmod -a 2.4.19-xfs
Put new kernel into place

- The generated kernel is going to be:
  /path/to/linux/source/arch/s390/boot/image
  AKA
  arch/s390/boot/image
- Copy the image file to /boot/
- Copy the System.map file to /boot/
  (located in the top-level source directory)
- Copy the .config file to /boot/
  (give it a name like config-2.4.26[-something] )
Regenerate the initrd

- Newer versions of SUSE and Red Hat use an initial ramdisk to hold driver modules
- Updating the kernel and/or kernel modules requires that the initrd be re-created
- The command that does this is “mkinitrd.”
  - Read the man page for this to understand what it does.
  - Look inside the initrd to see what’s in the old one, versus the new one.
  - Look at http://linuxvm.org/Info/HOWTOs/mkinitrd-notes.html
Update /etc/zipl.conf

- Review the contents of /etc/zipl.conf
- If you need to make a change, do so
  - Correct kernel
  - Correct default kernel
  - Correct DASD volume to write the kernel
  - Correct kernel parameters specified
Re-run zipl

- If you use /etc/zipl.conf, just type in “zipl”
- If you don’t use /etc/zipl.conf, then you’ll have to specify all the parameters:
  - zipl -t /boot -i /boot/image-2.4.26 -p /boot/parmfile -r /boot/ramdisk
- Make sure you get messages similar to this:
  Building bootmap ‘./bootmap’
  Adding IPL section
    kernel image......: image at 0x10000
    kernel parmline...: 'dasd=300-305,400 root=/dev/dasda1 ro noinitrd' at 0x1000
  Preparing boot device: dasda (0300).
  Done.
Take the system down

- `shutdown -h now`
- `shutdown -h 23:59`
- Whatever your site’s change management dictates.
Boot from the new kernel

• In an LPAR - from the HMC
• From z/VM - ipl devno clear
• How do you know what to specify for the boot device number?
  • From the /boot directory:
    df -h .
    grep dasd? /proc/dasd/devices
    First number is the device number
Back off to the old kernel

How do you do that, when you just over-wrote your old kernel information?

- You need multiple DASD volumes/minidisks (not LVM or RAID)
- Create a boot directory (or some other name) in each file system
- Copy the files from /boot, and your new kernel, etc.
- Re-run zipl from that directory or add entries to /etc/zipl.conf and change your default

```
# df -h

Filesystem   Size  Used  Avail  Use% Mounted on
/dev/dasda1   2.3G   348M  1.8G   17%  /
/dev/dasdb1   2.3G   1.3G  848M   61%  /usr
```