Operating System - Linux

- What is Linux?
- Why Linux?
  - Most popular Open Source operating system
  - Marketed commercially now
  - Easily modified and tweaked
  - Pool of experienced users
  - Easily administered remotely
  - Can be trimmed down (to 600KB) – reduce potential for bugs
  - Supports many processor architectures (Alpha, IA32, IA64, PowerPC, Opteron)
- Does it need to be modified for HPC? No.

Kernel v Distribution

- Linux is kernel which controls hardware, multitasking, virtual memory, shared libraries, demand laoding, shared copy-on-write executables, TCP/IP, file systems
- Distribution usually includes installer and also includes many other public domain programs
  - RedHat, SuSe, Fedora, Mandrake, SlackWare,
- Also integrated hardware/software Beowulf solutions based on one of these
- Beowulf setup systems e.g. OSCAR, NPACI Rocks

GPL v Open Source

- Modifications of GPLed software must not be distributed as binary only . Source must be made available
  - Linux is GPLed
- Open Source software which is not GPLed may be modified and sold as binary only code.
  - Mozilla, X-windows, BSD, MPICH

Linux Distributions

Red Hat	www.redhat.com
SuSe	www.suse.com
Mandrake	www.mandrake.com
Debian	www.debian.org
SlackWare	www.slackware.com
TurboLinux	www.turbolinux.com
Connectiva	www.connectiva.com
Gentoo	www.gentoo.org
Fedora	www.fedora.us fedora.redhat.com

Which Distribution for a Cluster ?

- Local Familiarity
- Language Support
- Bundled Software/Hardware
- Cluster versions
  - OSCAR, NPACI Rocks
  - Essentially diffs of standard distributions
- LIcensing

Version Numbers

- Kernel
  - Linus Torvalds and Core Team (Donald Becker, Alan Cox, Stephen Tweedie, David Miller)
  - Contributions sent in may be incorporated
- Distribution version number and Kernel version number not related
- Kernel versions
  - Stable (even minor numbers) 2.2, 2.4
  - Development (odd minor numbers) 2.1, 2.5
- Distributions choose version numbers as they please
  - May also modify basic kernels
  - If so, generic upgrades will not work

Tracking down kernel/driver issues

- Read the documentation
  - HOWTO documents in /usr/doc/HOWTO
- Web surf (start at Google!!)
- Consult local Linux users
- Read mailing lists & search for your topic
  - Archives like marc.theaimsgroups.com
- If you narrow down mail the *appropriate* mailing groups
- As a last resort look at source code and mail author

Compiling the Linux Kernel

- /proc – interface to kernel data structures
  - ls –l /proc/version ; cat /proc/version
- Cd /usr/src – this is often where the kernel source is (if you selected “kernel source” when you installed)
  - ls –ld linux - see it is a symbolic link
- If you are lucky can compile with
  - make clean; make bzImage
- Try
  - ls –l /usr/src/linux/arch/i386/boot/bzImage

Loadable Kernel Modules

- Dynamic way to extend kernel functionality
  - Don't retain in memory, don't require kernel recompile
  - Helps to keep kernel small and aids stability
- Modules for device drivers, file systems, special features
- May get 500 or more loadable modules in a distribution

Slimming the Kernel

- Need to re-configure see README in kernel source directory
  - The graphic version popular (make xconfig)
- Start slow, remove a few features, recompile, test
- Think server
  - Remove things like radio, sound, IrDA, ISDN, ARCnet. Other networks not used, USB (if don't have USB keyboard/mouse), joystick, telephony
- Optimize for CPU
  - Compile to use most recent instruction set your processor supports

Slimming the Kernel (ctd.)

- Optimize for number of processors
  - If only one CPU remove SMP support
- Remove firewall or DoS protection
  - Intensive message passing can be mistaken for DoS??
  - Reduces overhead
- Could also compile all modules in and remove loadable module support
- Could reduce from 1.5MB with 10MB of loadable modules to 600Kb with no loadable modules

Possibly Worth Supporting

- NFS – for small clusters
- Serial console
- Kernel IP configuration – get IP address using BOOTP or DHCP
- NFS root – supports diskless booting by allowing mounting of root file systems
- Special high performance network drivers – Gigabit Ethernet, Myrinet
- A file system

Network Booting

- Allows kernel to be loaded from NAS (network attached storage)
- Need specialized BIOS or network adapter
- Most common standard Intel PXE 2.0
  - Firmware boot code requests address and kernel from NAS and gets kernel with TFTP
  - TFTP not scalable
    - Need to limit number of nodes booting or use multiple TFTP servers and segregate Ethernet collision domains

Diskless Operation

- Why?
  - Security reasons
  - If need to change kernels/distributions frequently
  - Only need to maintain one image
- See Diskless-HOWTO and Diskless-root-NFS-HOWTO
- Need NFS root to mount other needed configuration files (/etc/passwd etc) and dynamic libraries
- NFS is not scalable for large clusters (see later)

Downloading and Compiling Kernel

- Download from www.kernel,org
- Read documentation – may need to download other components (e.g. libc)
- Distribution kernels may have mods from stock kernel e.g. device drivers, tuning, etc
  - Can go to entirely generic
  - Can try to download from distribution company
  - Can try to add mods to stock kernel

Linux File Systems

- Default is EXT2 (extended file system version 2)
- EXT2 is not a journalling file system, one where writes ensure that file system is always or can always be put in a consistent state – avoids the need for fsck
- Slightly slower – must write “journal” to disk first, which will enable restoration of consistent state
- So depends on whether want optimum disk performance on local nodes
- Journalling systems: EXT3, ReiserFS (SuSe, better for small files/large dirs) , IBM JFS, SGI XFS (optimized for large block writes from virtual memory)

Networked & Distributed File Systems

- Local file systems for scratch data
- Networked file system for sharing data
  - NFS (mounts file system over IP)
    - Problems: scalability and synchronization
    - Performance seriously degrades for > 64 nodes
    - Should not write files in expectation they will be available to other nodes
- Solutions are still experimental
  - E.g. PVFS

Pruning the Node

- Start from server option of installation
- Prune applications automatically started by inetd/xinetd and init.d
- Inetd/xinetd superserver spawns programs to serve requests on sets of ports (see /etc/inetd.conf & /etc/services or /etc/xinetd.d)
- Can eliminate services not needed
- In fact in secure systems where ssh run as daemon may be able to eliminate inetd process itself

Boot Scripts

- /etc/rc.d/init.d scripts run at boot time that often run daemons
- Run as enter or leave run level
- Some scripts only initialize hardware or change settings
- Not all scripts run – which are run at each level can be seen by
  - chkconfig –list | grep '3;on'
- No need for lpd, mysql, httpd, named, sendmail, etc
- Normally no need for X windows
- Normally run level 4 is the highest run level on a compute node

Other Processes

- cron scripts
- slocate for indexing file system
- Use to see process and memory they use

```
ps –eo pid,pcpu,sz,vsize,user,fname –sort=vsize
```

Scalable Services

- OS rely on network for services such as time and DNS
- Can cause performance bottlenecks
  - DNS lookups could access a campus server
  - TCP might do reverse DNS lookup per TCP connection
  - NFS, NIS similarly don't scale well

Virtual Memory Problems

- Demand paged virtual memory usually incurs small performance penalty
- But can be large
- Can lead to mystifying anomalies
  - Extra daemon on nodes causes swapping
- Consider server with 256MB memory
  - Program with 300MB memory usage causes massive swapping (377,093 page faults) and takes 5 minutes
  - With 150MB array only takes 0.5sec and 105 page faults
- May be able to tune to improve performance on clusters
  - Virtual memory/cache locality

Excessive Paging Example

```
#define MEGABYTES 300
main(){
    int *x, *p, t=1, I, numints=MEGABYTES*1024*1024/sizeof(int);
    x = (int *) malloc(numints*sizeof(int));
    if (!x) { printf"insufficient memory\n"); exit(1); }
    for (i=1; i<=5; i++) {
        printf(Loop %d\n",i);
        for(p=x; p<x+numints-1; p+=1024) {
            *p = *p + t;
        }
    }
}
```


TCP messaging

- Normally tuned for general purpose computing
- In clusters, short low-latency and very long messages common
- 2.2 kernels needed tweaks to stack to improve performance
- See **Paul A. Farrell, Hong Ong, Communication Performance over a Gigabit Network, http://discov.cs.kent.edu/publications/2000/ipccc2000.pdf** for some of this
- 2.4 kernels generally OK
- May need to tune for high speed networks like Myrinet
- Browse the Beowulf mailing lists

Final Tuning with /proc

- Probably not much performance improvement unless something is wrong
- See www.linuxhq.com & linuxperf.nl.linux.org if you want to try
- Networking – try
  - cat /proc/net/dev or run /sbin/ifconfig
  - Check using the correct interface
  - Look at collisions, errs, dropped, frame
    - If dropped growing by few packets per sec problem
- Tunable parameters are in /proc/sys/net
  - tcp_sack, tcp_window_scaling etc

Other /proc tuning

- Memory
  - cat /proc/meminfo to see parameters
  - Tune in /proc/sys/vm
- File system /proc/sys/fs
- Harddisk
  - /sbin/hdparam
- Kernel basics /proc/sys/kernel
  - E.g. /proc/sys/kernel/shmem is maximum size of shared memory segments