Linux Clusters for High-Performance Computing: An Introduction

Jim Phillips, Tim Skirvin

Outline

- Why and why not clusters?
- Consider your…
  – Users
  – Application
  – Budget
  – Environment
  – Hardware
  – System Software

HPC vs High-Availability

- There are two major types of Linux clusters:
  - High-Performance Computing
    - Multiple computers running a single job for increased performance
  - High-Availability
    - Multiple computers running the same job for increased reliability
  - We will be talking about the former!

Why Clusters?

- Cheap alternative to “big iron”
- Local development platform for “big iron” code
- Built to task (buy only what you need)
- Built from COTS components
- Runs COTS software (Linux/MPI)
- Lower yearly maintenance costs
- Single failure does not take down entire facility
- Re-deploy as desktops or “throw away”

Why Not Clusters?

- Non-parallelizable or tightly coupled application
- Cost of porting large existing codebase too high
- No source code for application
- No local expertise (don’t know Unix)
- No vendor hand holding
- Massive I/O or memory requirements

Know Your Users

- Who are you building the cluster for?
  - Yourself and two grad students?
  - Yourself and twenty grad students?
  - Your entire department or university?
- Are they clueless, competitive, or malicious?
- How will you to allocate resources among them?
- Will they expect an existing infrastructure?
- How well will they tolerate system downtimes?

Your Users' Goals

- Do you want increased throughput?
  - Large number of queued serial jobs.
  - Standard applications, no changes needed.
- Or decreased turnaround time?
  - Small number of highly parallel jobs.
  - Parallelized applications, changes required.

Your Application

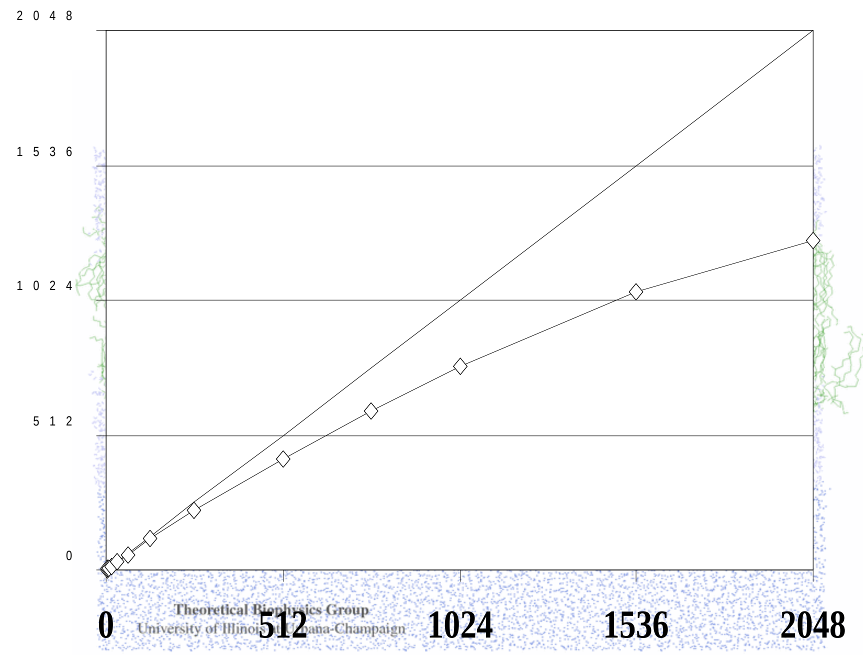
- The best benchmark for making decisions is your application running your dataset.
- Designing a cluster is about trade-offs.
  - Your application determines your choices.
  - No supercomputer runs everything well either.
- Never buy hardware until the application is parallelized, ported, tested, and debugged.

Your Application: Serial Performance

- How much memory do you need?
- Have you tried profiling and tuning?
- What does the program spend time doing?
  - Floating point or integer and logic operations?
  - Using data in cache or from main memory?
  - Many or few operations per memory access?
- Run benchmarks on many platforms.

Your Application: Parallel Performance

- How much memory per node?
- How would it scale on an ideal machine?
- How is scaling affected by:
  - Latency (time needed for small messages)?
  - Bandwidth (time per byte for large messages)?
  - Multiprocessor nodes?
- How fast do you need to run?



NAMD
Scalable Molecular Dynamics

Budget

- Figure out how much money you have to spend.
- Don’t spend money on problems you won’t have.
  - Design the system to just run your application.
- Never solve problems you can’t afford to have.
  - Fast network on 20 nodes or slower on 100?
- Don’t buy the hardware until…
  - The application is ported, tested, and debugged.
  - The science is ready to run.

Environment

- The cluster needs somewhere to live.
  - You won't want it in your office.
  - Not even in your grad student's office.
- Cluster needs:
  - Space (keep the fire martial happy).
  - Power
  - Cooling

Environment: Space

- Rack or shelve systems to save space
  - 36” x 18” shelves ($180 from C-Stores)
    - 16 typical PC-style cases
    - 12 full-size PC cases
    - Wheels are nice and don’t cost much more
    - Watch for tipping!
  - Multiprocessor systems save space
  - Rack mount cases are smaller but expensive

Environment: Power

- Make sure you have enough power.
  - Kill-A-Watt
    - $30 at ThinkGeek
  - 1.3Ghz Athlon draws 183 VA at full load
    - Newer systems draw more; measure for yourself!
    - More efficient power supplies help
  - Wall circuits typically supply about 20 Amps
    - Around 12 PCs @ 183VA max (8-10 for safety)

Environment: Power Factor

- Always test your power under load
- More efficient power supplies do help!

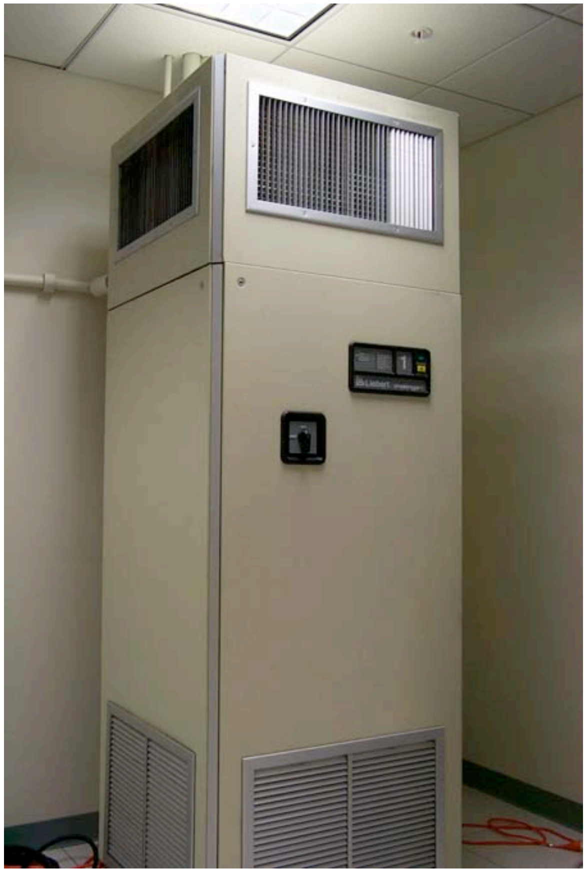
Athlon 1333 (Idle)	1.25A	98W	137VA	PF 0.71
Athlon 1333 (load)	1.67A	139W	183VA	PF 0.76
Dual Athlon MP 2600+	2.89A	246W	319VA	PF 0.77
Dual Xeon 2.8GHz	2.44A	266W	270VA	PF 0.985

Environment: Uninterruptable Power Systems

- 5kVA UPS ($3,000)
  - Holds 24 PCs @183VA (safely)
  - Rackmount or stand-alone
  - Will need to work out building power to them
  - Larger/smaller UPS systems are available
  - May not need UPS for all systems, just root node

Environment: Cooling

- Building AC will only get you so far
- Make sure you have enough cooling.
  - One PC @183VA puts out ~600 BTU of heat.
  - 1 ton of AC = 12,000 BTUs = ~3500 Watts
  - Can run ~20 CPUs per ton of AC



Hardware

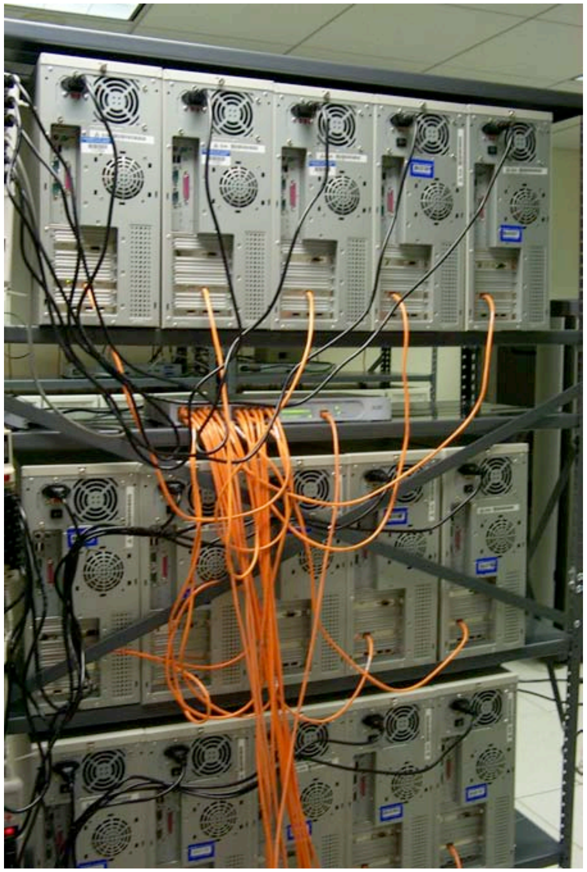
- Many important decisions to make
- Keep application performance, users, environment, local expertise, and budget in mind
- An exercise in systems integration, making many separate components work well as a unit
- A reliable but slightly slower cluster is better than a fast but non-functioning cluster

Hardware: Computers

- Benchmark a "demo" system first!
- Buy identical computers
- Can be recycled as desktops
  - CD-ROMs and hard drives may still be a good idea.
  - Don't bother with a good video card; by the time you recycle them you'll want something better anyway.

Hardware: Networking (1)

- Latency
- Bandwidth
- Bisection bandwidth of finished cluster
- SMP performance and compatibility?



Hardware: Networking (2)

- Two main options:
  - Gigabit Ethernet – cheap ($100-200/node), universally supported and tested, cheap commodity switches up to 48 ports.
    - 24-port switches seem the best bang-for-buck
  - Special interconnects:
    - Myrinet – very expensive ($thousands per node), very low latency, logarithmic cost model for very large clusters.
    - Infiniband – similar, less common, not as well supported.

Hardware: Gigabit Ethernet (1)

- The only choice for low-cost clusters up to 48 processors.
- 24-port switch allows:
  - 24 single nodes with 32 bit 33 MHz cards
  - 24 dual nodes with 64 bit 66 MHz cards

Hardware: Gigabit Ethernet (2)

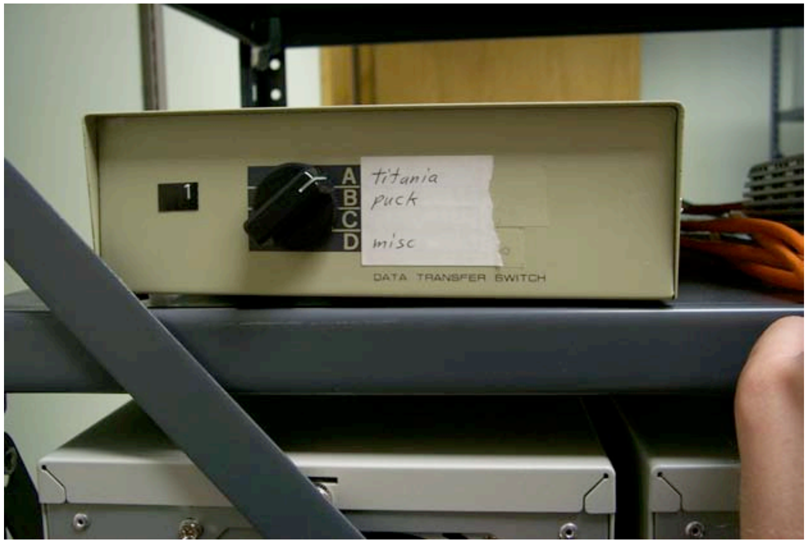
- Jumbo frames:
  - Extend standard ethernet maximum transmit unit (MTU) from 1500 to 9000
  - More data per packet, fewer packets, reduced overhead, lower processor utilization.
  - Requires managed switch to transmit packets.
  - Incompatible with non-jumbo traffic.
  - Probably not worth the hassle.

Hardware: Gigabit Ethernet (3)

- Sample prices (from cdwg.com)
  - 24-port switches
    - SMC EZSwitch SMCGS24 unmanaged $ 374.00
    - 3Com Baseline 2824 unmanaged $ 429.59
    - ProCurve 2724 managed $1,202.51
  - 48-port switches
    - SMC TigerSwitch SMC6752AL2 unmanaged $ 656.00
    - 3Com SuperStack 3848 managed $3,185.50
    - ProCurve 2848 managed $3,301.29
  - Network Cards
    - Most are built-in with current architectures
    - Can buy new cards for $25-60

Hardware: Other Components

- Filtered Power (Isobar, Data Shield, etc)
- Network Cables: buy good ones, you'll save debugging time later
- *If a cable is at all questionable, throw it away!*
- Power Cables
- Monitor
- Video/Keyboard Cables



System Software

- "Linux" is just a starting point.
  - Operating system,
  - Libraries - message passing, numerical
  - Compilers
  - Queuing Systems
- Performance
- Stability
- System security
- Existing infrastructure considerations

System Software: Operating System (1)

- Clusters have special needs, use something appropriate for the application, hardware, and that is easily clusterable
- Security on a cluster can be nightmare if not planned for at the outset
- Any annoying management or reliability issues get hugely multiplied in a cluster environment

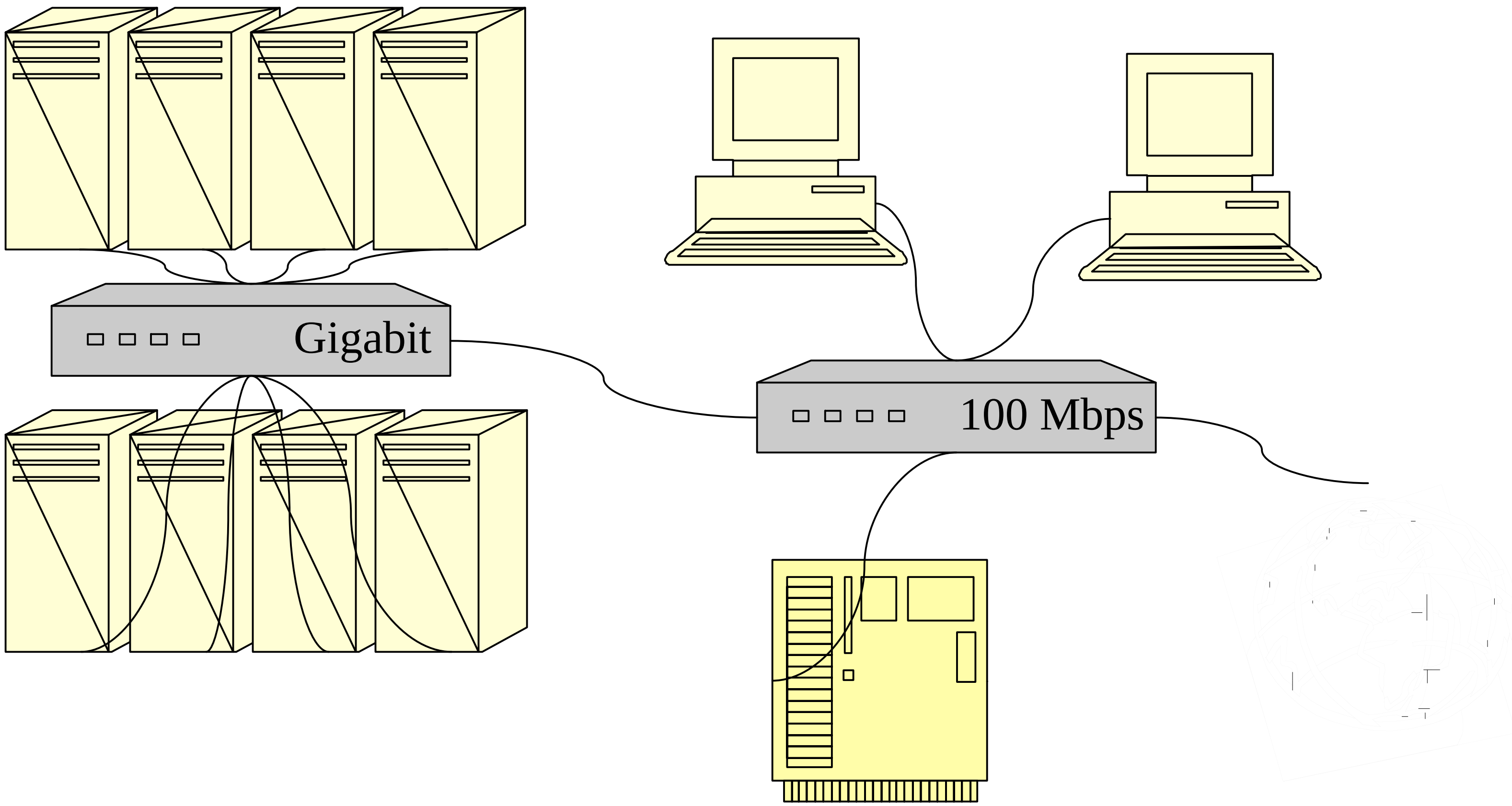
System Software: Operating System (2)

- SMP Nodes:
  - Does kernel TCP stack scale?
  - Is message passing system multithreaded?
  - Does kernel scale for system calls made by intended set of applications?
- Network Performance:
  - Optimized network drivers?
  - User-space message passing?
  - Eliminate unnecessary daemons, they destroy performance on large clusters (collective ops)

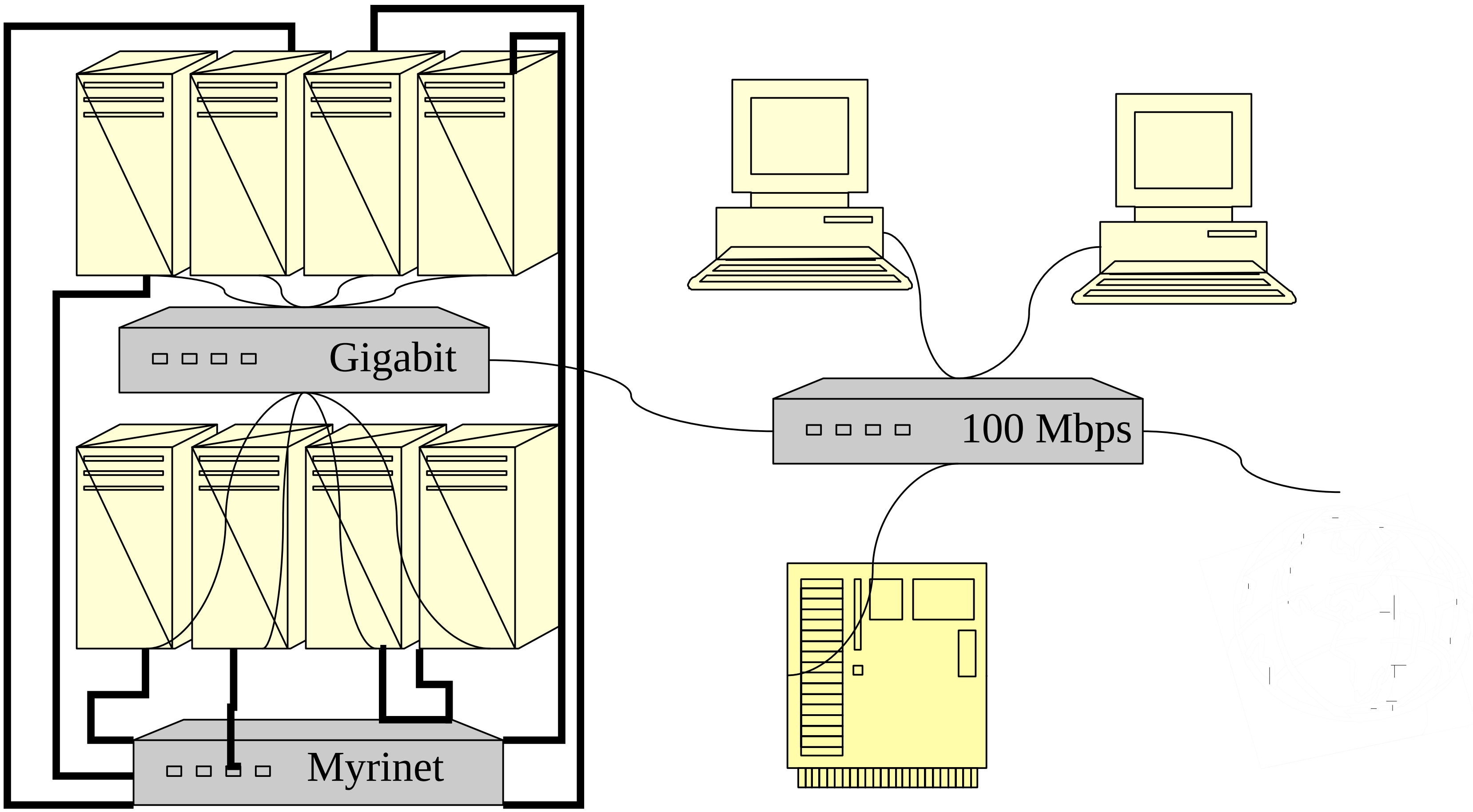
Software: Networking

- User-space message passing
  - Virtual interface architecture
  - Avoids per-message context switching between kernel mode and user mode, can reduce cache thrashing, etc.

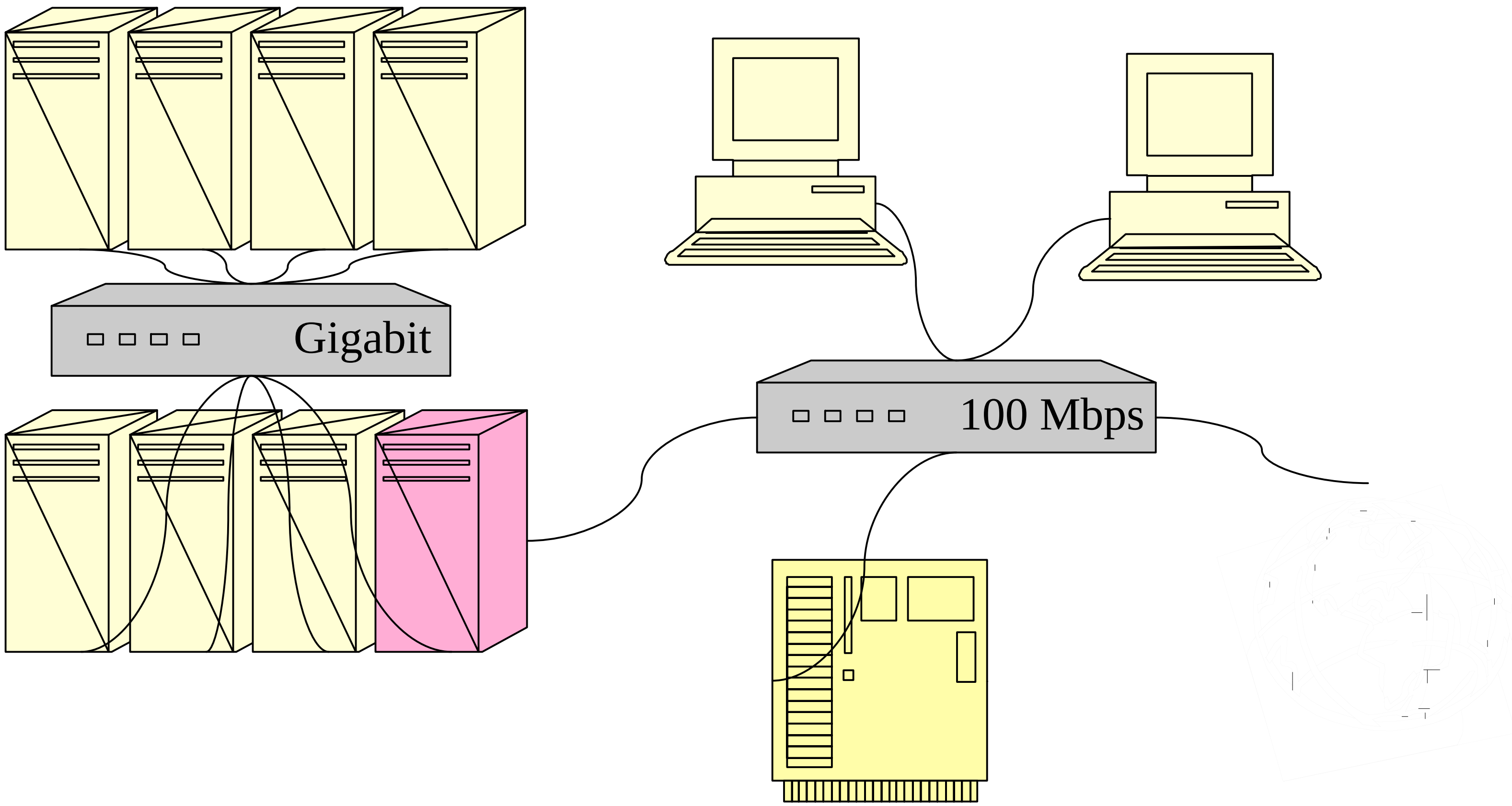
Network Architecture: Public



Network Architecture: Augmented



Network Architecture: Private



Scyld Beowulf / Clustermatic

- Single front-end master node:
  - Fully operational normal Linux installation.
  - Bproc patches incorporate slave nodes.
- Severely restricted slave nodes:
  - Minimum installation, downloaded at boot.
  - No daemons, users, logins, scripts, etc.
  - No access to NFS servers except for master.
  - Highly secure slave nodes as a result

Oscar/ROCKS

- Each node is a full Linux install
  - Offers access to a file system.
  - Software tools help manage these large numbers of machines.
  - Still more complicated than only maintaining one “master” node.
  - Better suited for running multiple jobs on a single cluster, vs one job on the whole cluster.

System Software: Compilers

- No point in buying fast hardware just to run poor performing executables
- Good compilers might provide 50-150% performance improvement
- May be cheaper to buy a $2,500 compiler license than to buy more compute nodes
- Benchmark real application with compiler, get an eval compiler license if necessary

System Software: Message Passing Libraries

- Usually dictated by application code
- Choose something that will work well with hardware, OS, and application
- User-space message passing?
- MPI: industry standard, many implementations by many vendors, as well as several free implementations
- Others: Charm++, BIP, Fast Messages

System Software: Numerical Libraries

- Can provide a huge performance boost over "Numerical Recipes" or in-house routines
- Typically hand-optimized for each platform
- When applications spend a large fraction of runtime in library code, it pays to buy a license for a highly tuned library
- Examples: BLAS, FFTW, Interval libraries

System Software: Batch Queueing

- Clusters, although cheaper than "big iron" are still expensive, so should be efficiently utilized
- The use of a batch queueing system can keep a cluster running jobs 24/7
- Things to consider:
  - Allocation of sub-clusters?
  - 1-CPU jobs on SMP nodes?
- Examples: Sun Grid Engine, PBS, Load Leveler