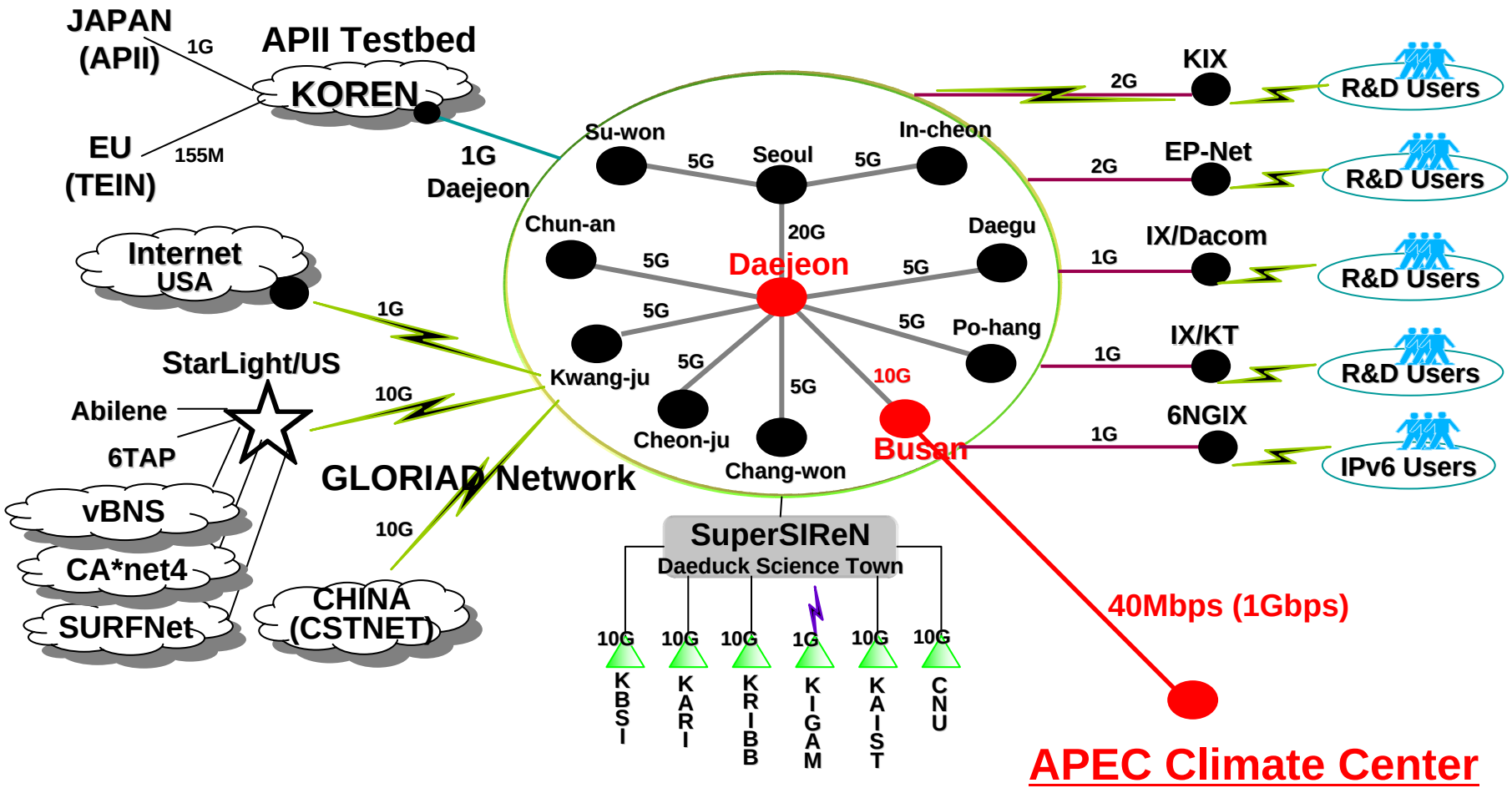
Introduction of APCC Computing System

Aug. 2007

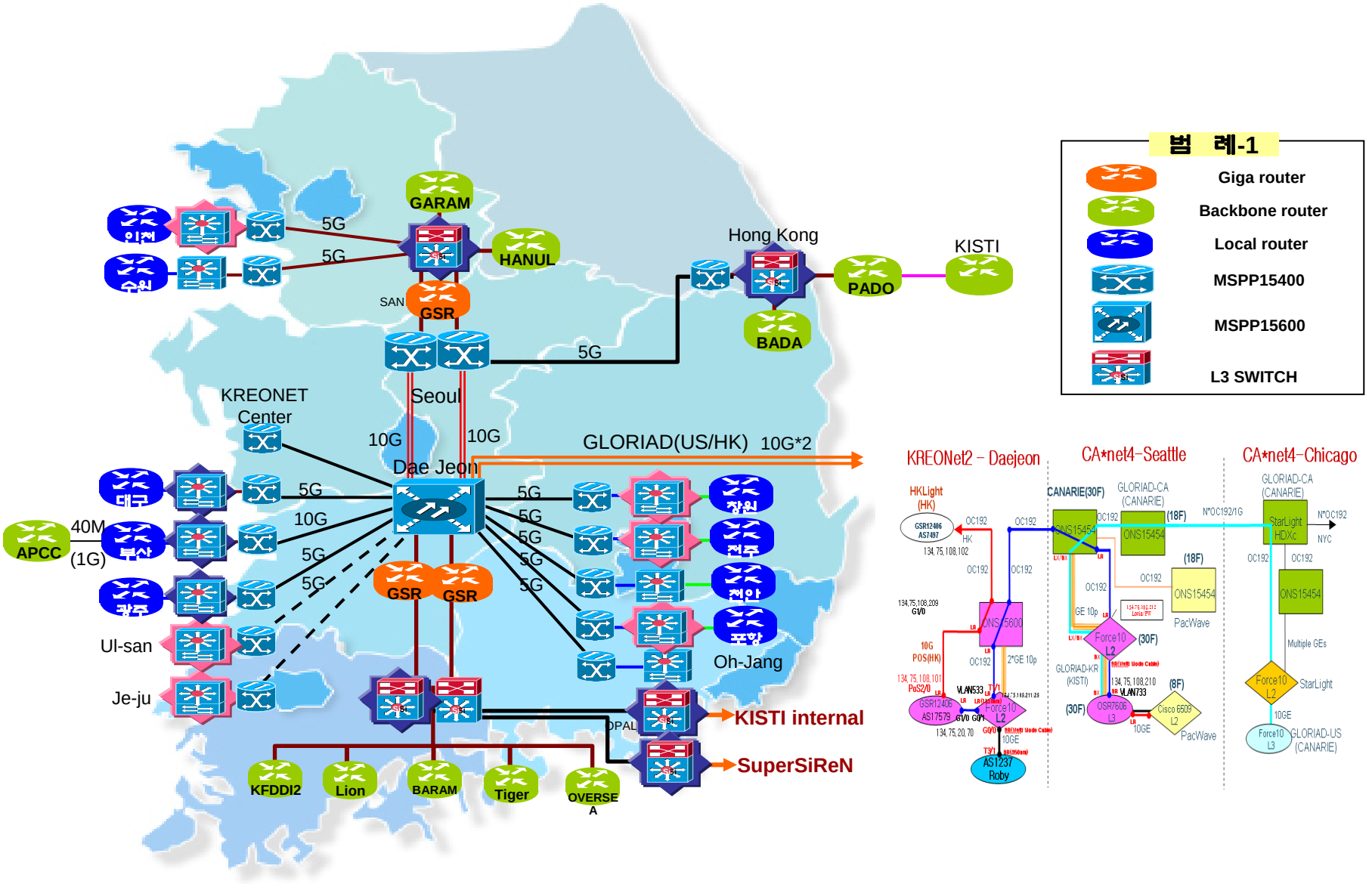
Hanse Yi

APCC R&D Network Topology

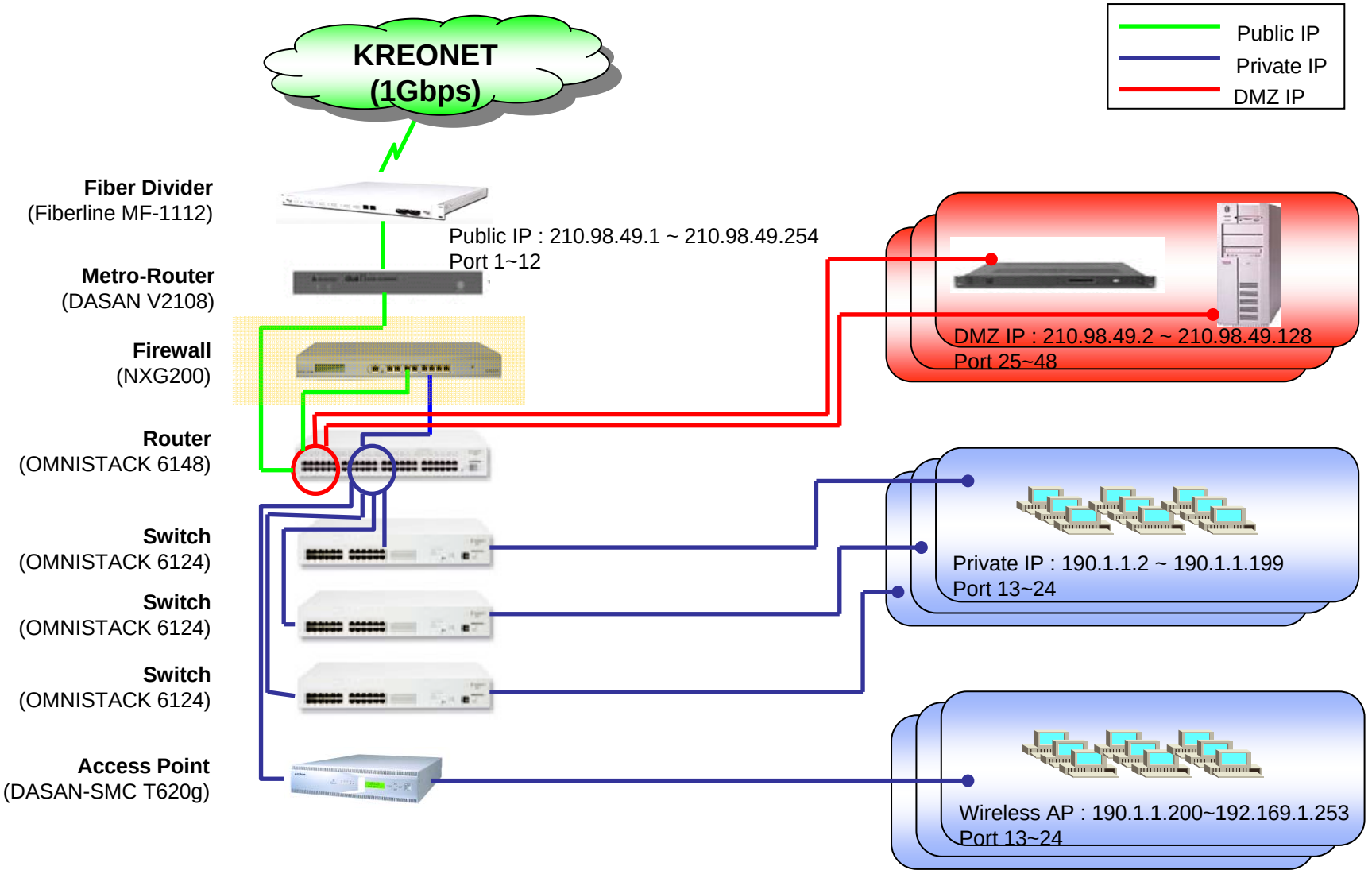
(based on KREONET)



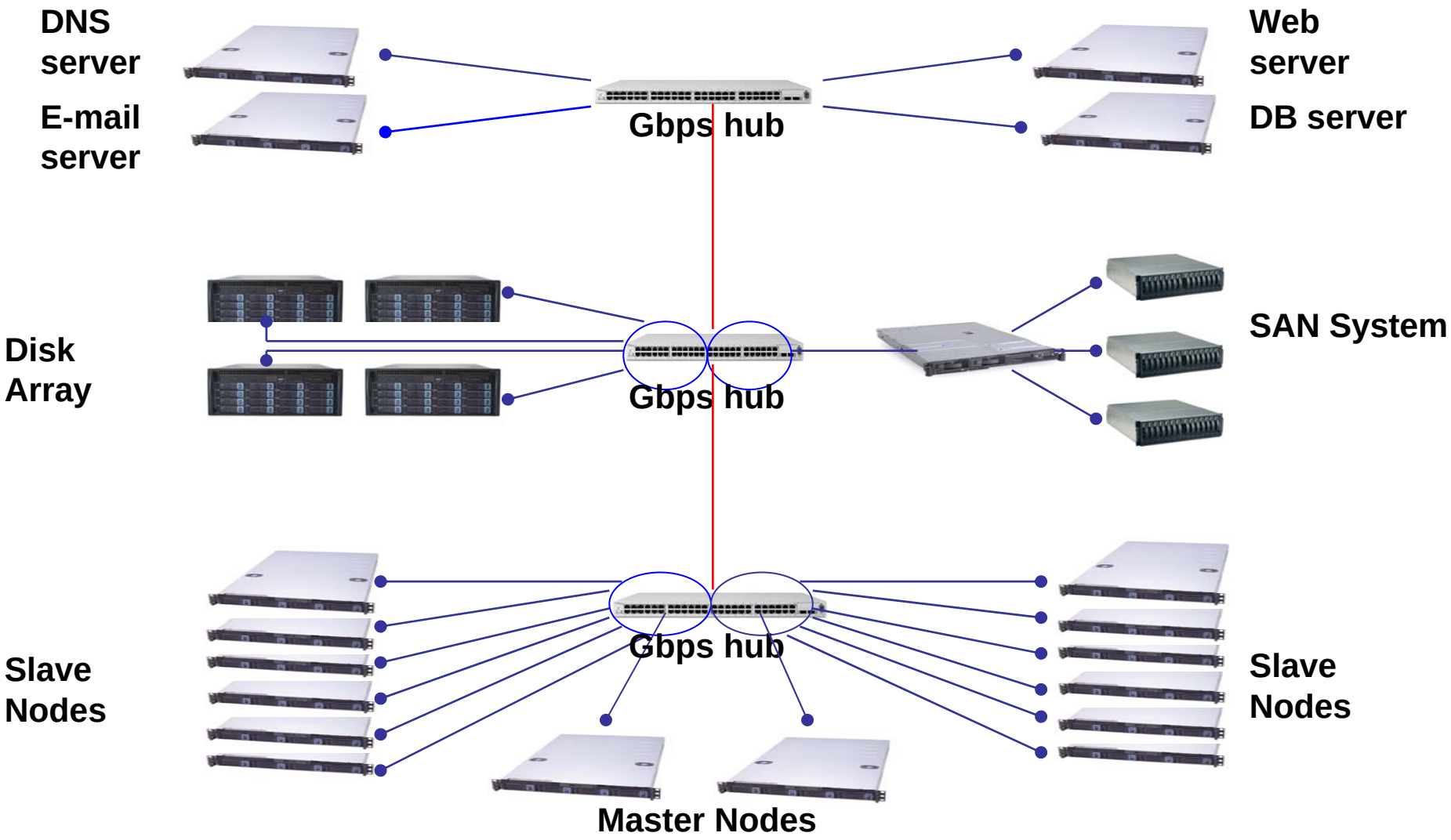
APCC KREONET local access backbone



Network Configuration



System Configuration

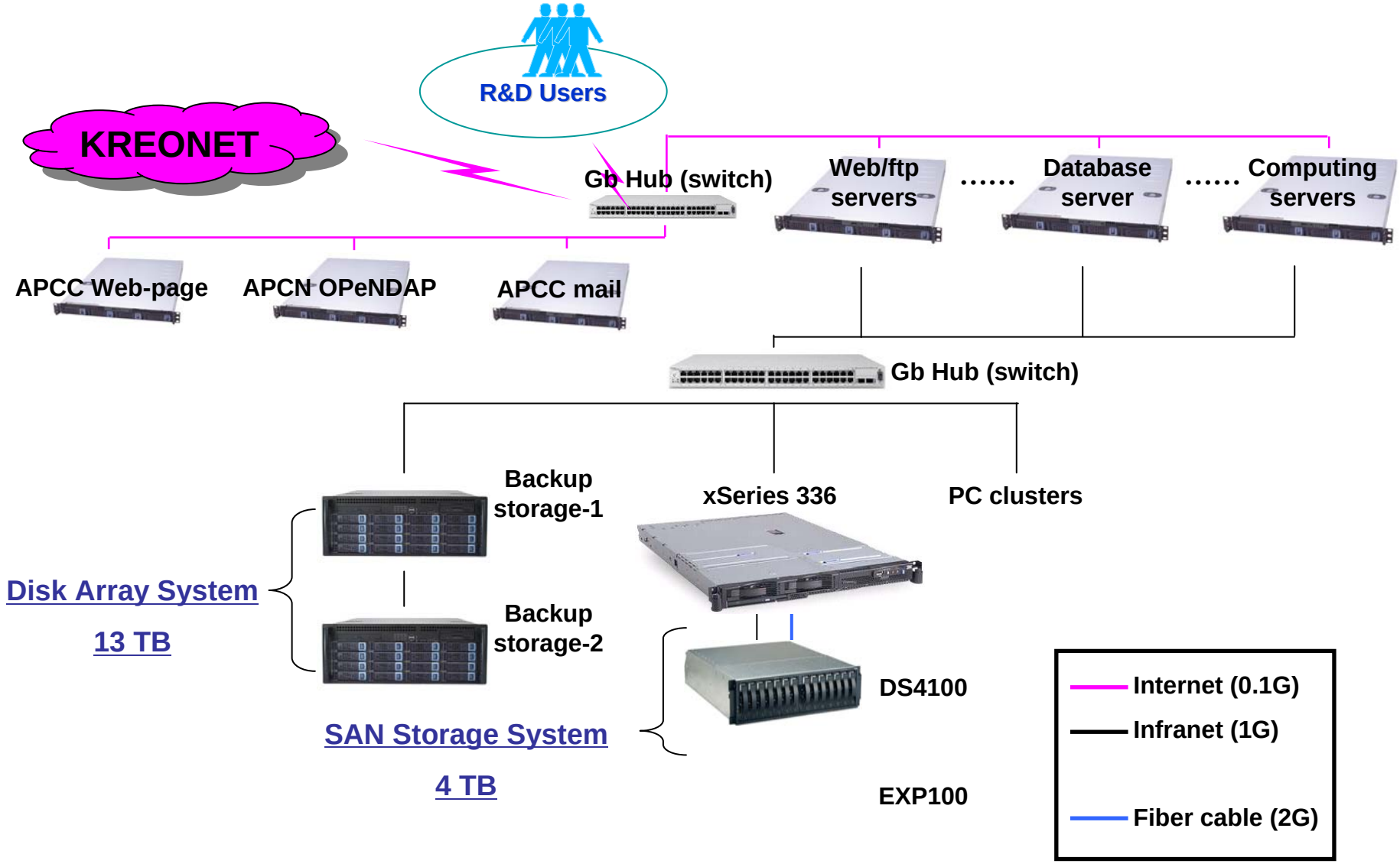


APCC user-group list of cluster

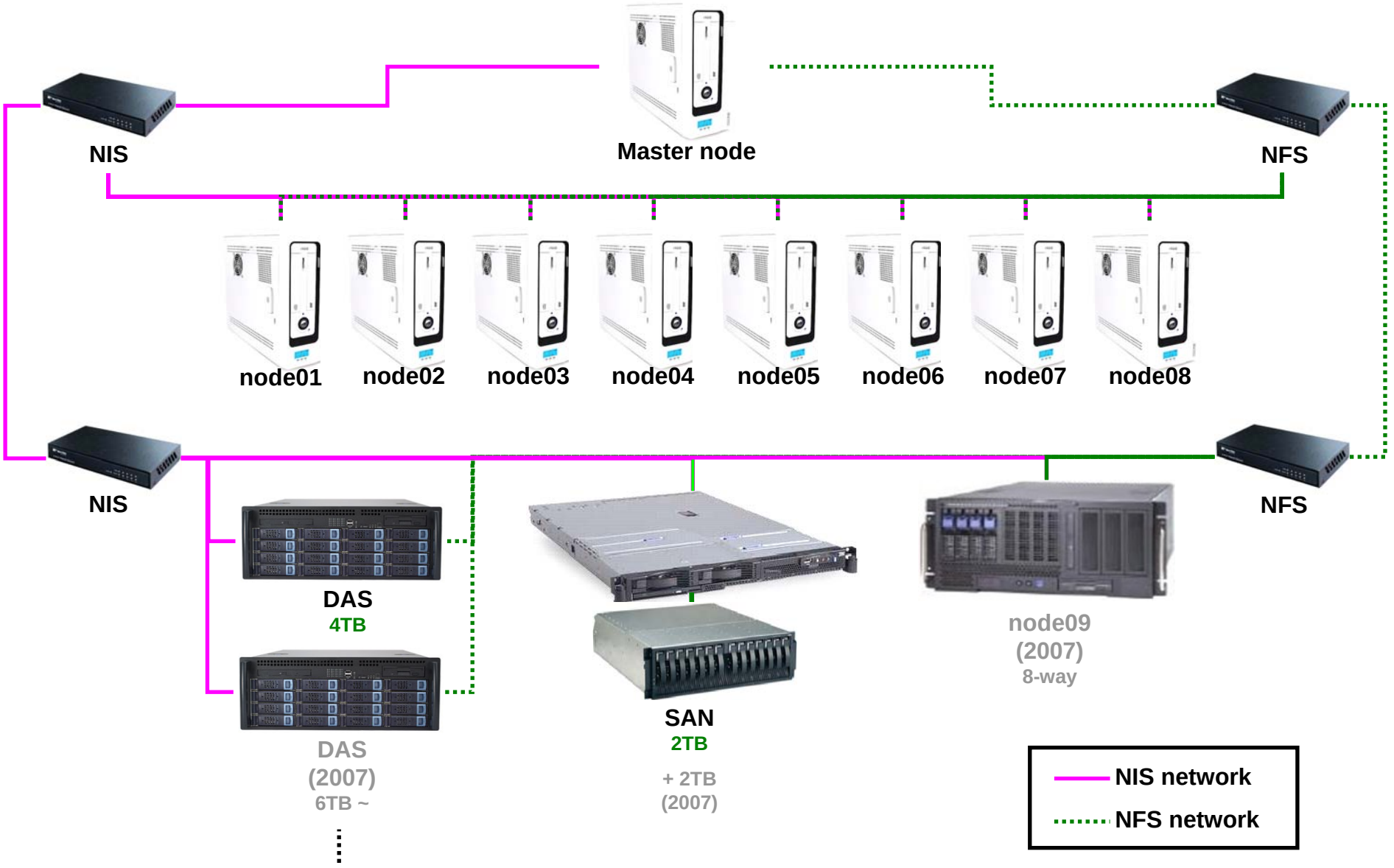
Group-A : Operational job accounts (oper, apcc, cis)

Group-B : Individual research accounts (others)

APCC System Configuration (~2007)



APCC PC cluster



Structure of directories (Now)

1. SAN

- Temporary disk
- Single-modelling data
- Link to the node01 ~ node08

nkeep01:

/apcc01

/apcc02

2. DAS

- Data backup
- Link to the node01 ~ node08

nkeep51:

/apcc51

/apcc52

3. Advaced DAS

- Data backup
- Link to the node09

nkeep52:

/apcc11

/apcc12

/apcc13

4. Advanced DAS

- Parallel-modelling data
- Data backup
- Link to the node09

nkeep11:

/apcc14

/apcc15

/apcc16

The directories of grey color will be added in 2007 yr, after introduction of new APCC system

Usage of directories

/apcc01 : common temporary file for all users

/apcc51 : data backup of group-A (only in nnode07 and 08)

/apcc52 : data backup of group-B (only in nnode07 and 08)

Structure of directories (future plan)

Usage of directories

/apcc01 : temporary data from group A
/apcc02 : temporary data from group B
/apcc11 : results of parallelized model
/apcc12 : data backup of parallelized model
/apcc13 : data backup of system & data team
/apcc14 : individual works-1
/apcc15 : individual works-2
/apcc15 : individual works-3
/apcc51 : backup disk of “/apcc01”
/apcc51 : backup disk of “/apcc01”

Examples of usage

1. Execution of executable file

```
cnode01 ~]$ cd /apcc01/~
cnode01 ~]$ a.out
```

2. Backup your files

```
cnode01 ~]$ rsh nnode07 (or 08)
cnode07 ~]$ cp –rf /apcc01/*.dat /apcc51/
cnode07 ~]$ rm –rf /apcc01/*.dat
```

Backup-job descriptions

Less data in temporary disk, more data in backup disk

1. Move your all data to “/apcc51” or “/apcc52” except operational job and your current job.

2. Remain the operational data produced by common account. (like oper and apcc)

3. Non-APCC user’s data will be moved by system administrator and deleted in temporary disk (“/apcc01”)

4. Please be empty your home directory.

5. Request the additional data backup to system administrator after your work done.

Spec-list of the 2nd supercomputer system (KMA)

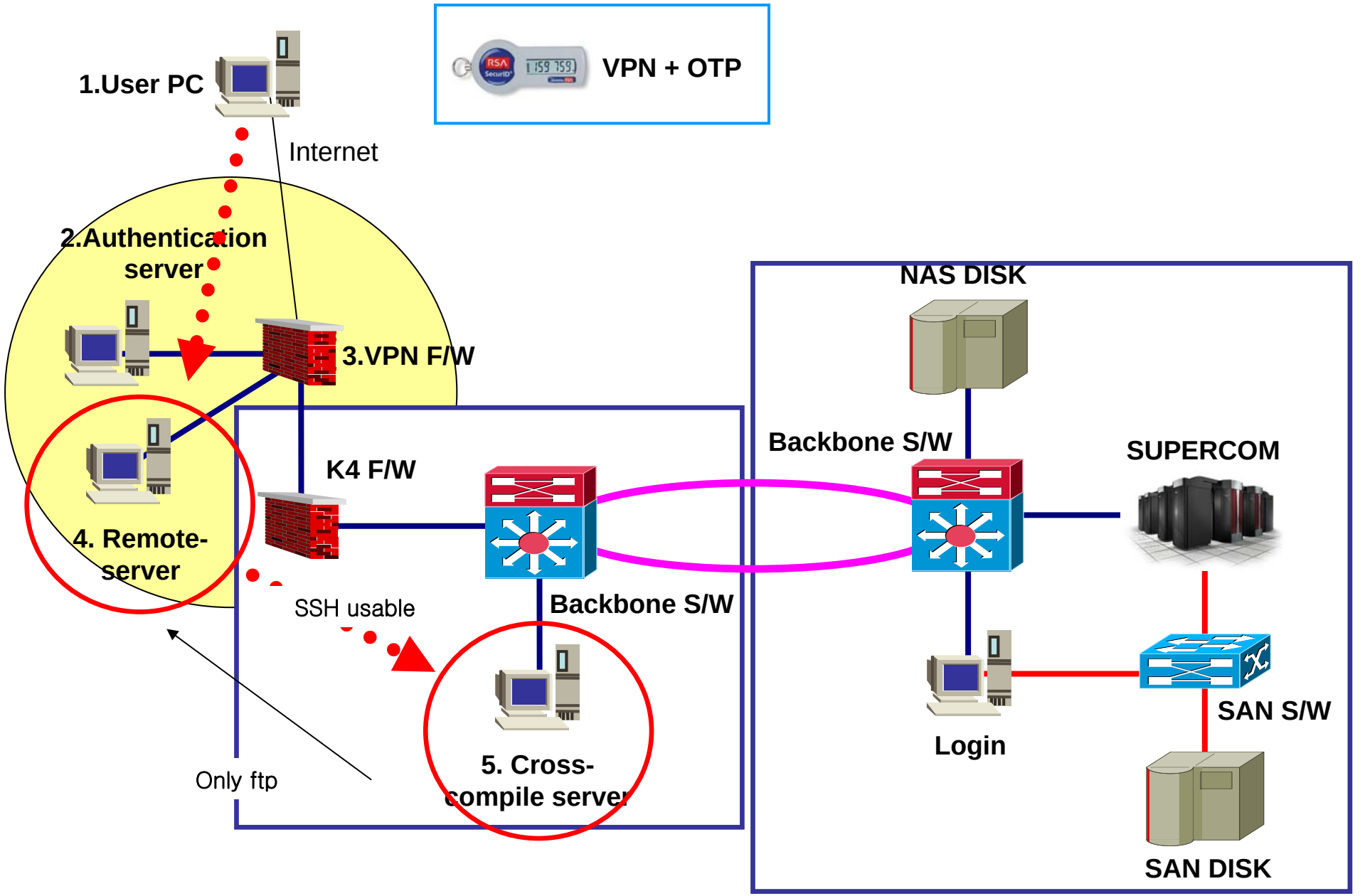
CRAY X1E

Tape drive (1PB)

Ref	CRAY X1E
Installation	2005. 11
CPU	1024 MSP
Peak performance	18.5 TF
Main Memory	4 TB
Capacity of DAS Disk	67 TB
Capacity of SAN Disk	21 TB
Capacity of Tape drive	1 PB
Functions	For NWP Operation Research & Development

KMA Supercomputer remote access system

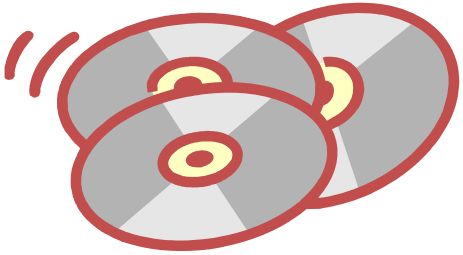
Remote Access step



Preparation

Internet available PC

Each PC

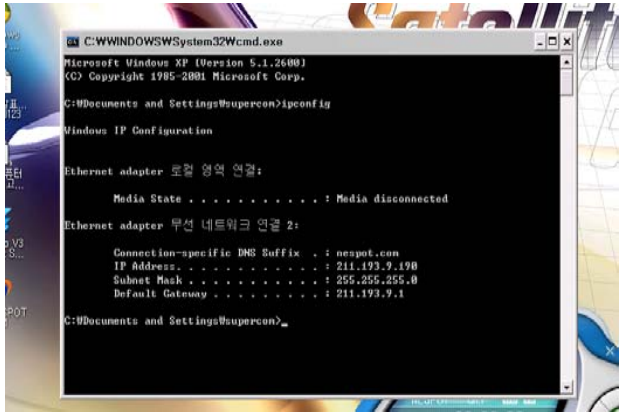


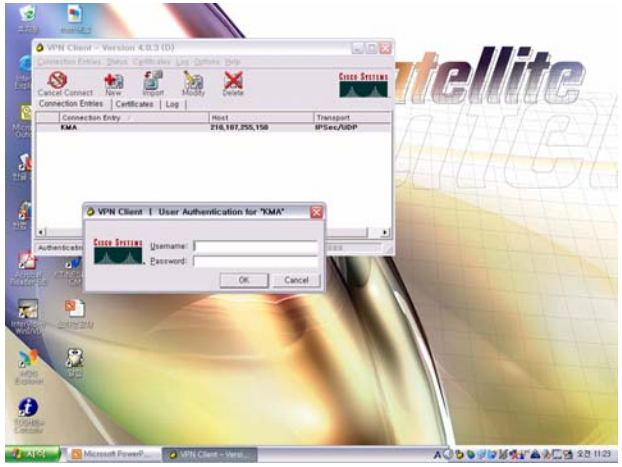
VPN client S/W

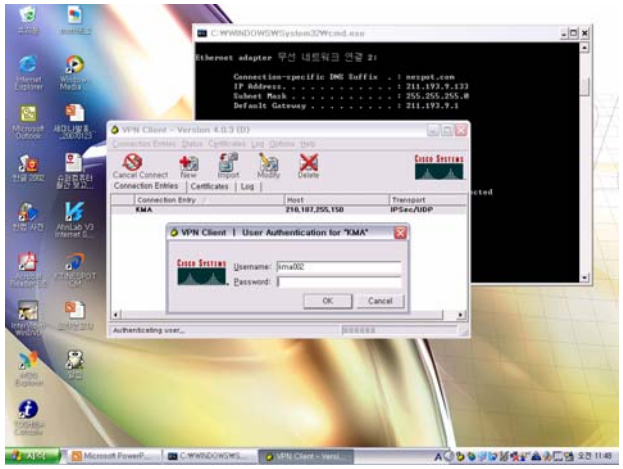
Supercom Home (super.kma.go.kr)
Download

OTP (One Time Password)

issue







Access step

1. PC network configuration before running VPN

   Click “connect” menue

2. Input “username” and “password”

3. Distributed OTP show the serial numbers,

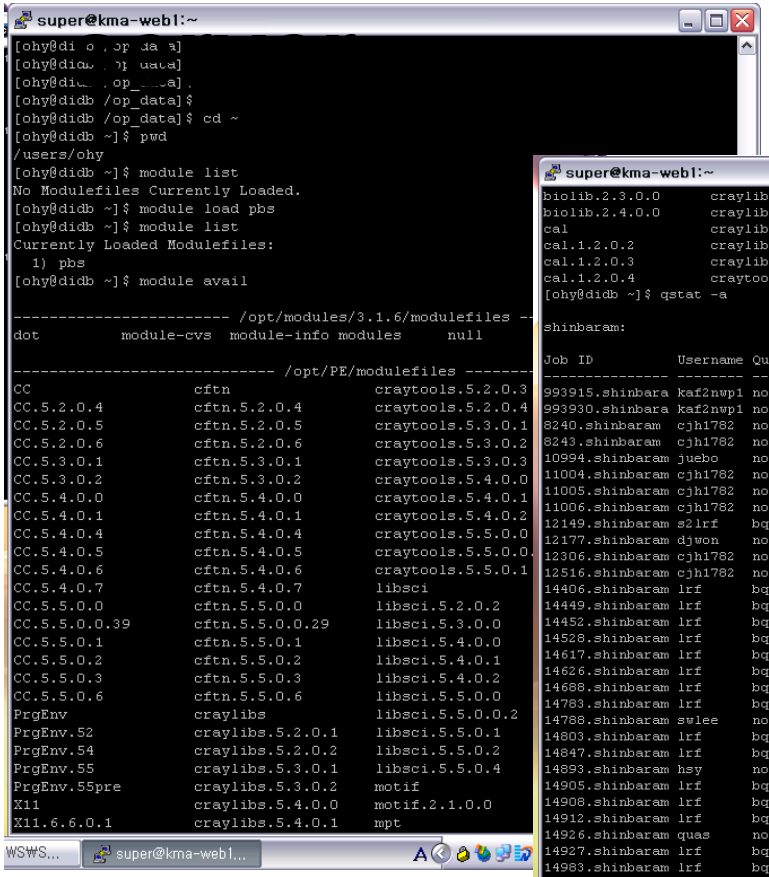
   each OTP match to username with 1 by 1.

4. This is example of some username’s usage called ‘kma002’

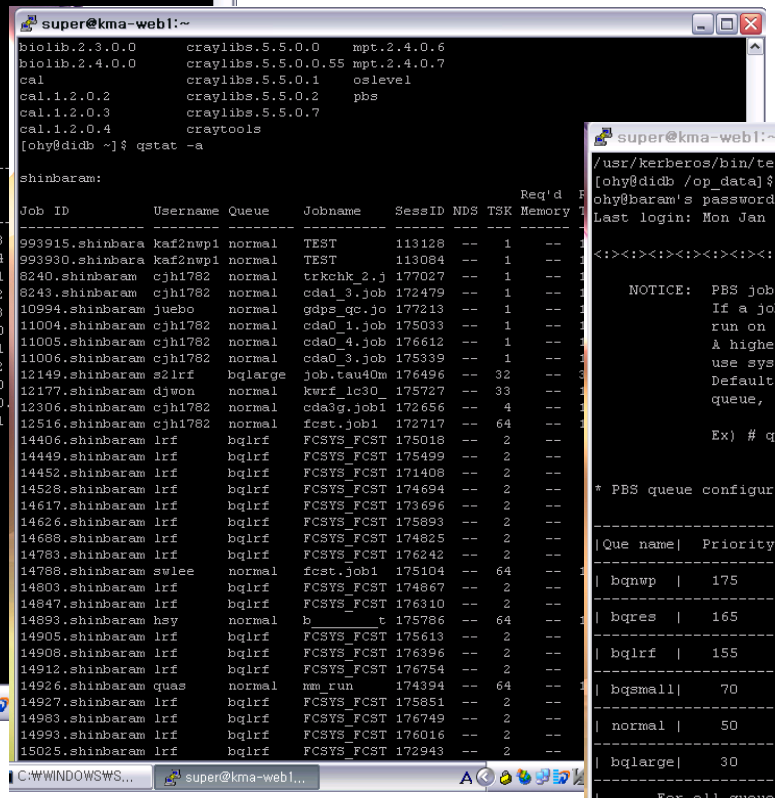
5. Input “kma002” in username’s empty space and

   input 6 number of OTP in passwd’s empty space

Function of Log-in [Cross-compiler]



CRAY Module loaded



Shinbaram's job monitoring page (PBS)

Shinbaram access page with operational account

Parallelized Computing for a Research Works

- To understand APCC clusters and outline of cluster

Aug. 2007

Hanse Yi

Resources : Beckman Institute, UIUC

Outline

- Why and why not clusters?

- Consider your…
  - Users
  - Application
  - Budget
  - Environment
  - Hardware
  - System Software

- Case study : APCC clusters

Why Clusters?

- Cheap alternative to “big iron”
- Local development platform for “big iron” code
- Built to task (buy only what you need)
- Built from free components
- Runs free software (Linux/MPI)
- Lower yearly maintenance costs
- 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?

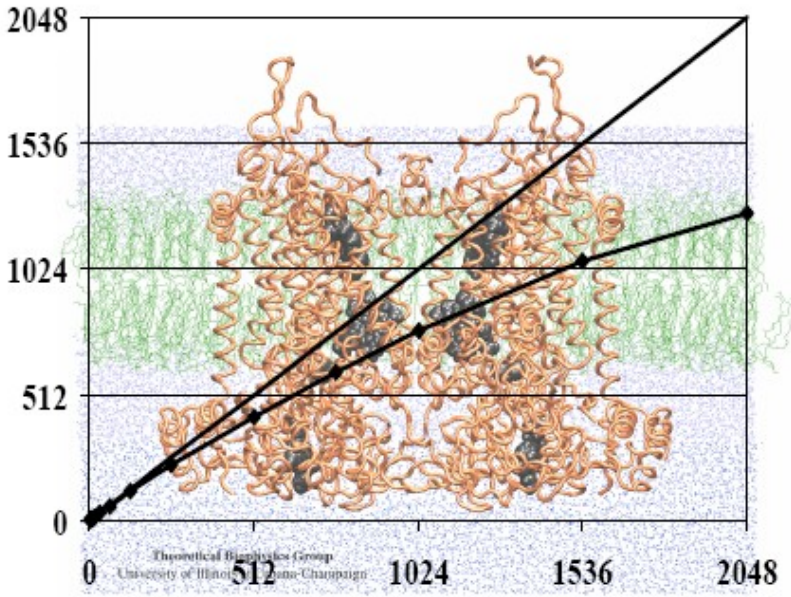
- Have you tried profiling and tuning?

- 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?



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 a grad student's office.

• Cluster needs:
– Space (keep the fire martial happy)
– Power
– Cooling

Environment: Space

• Rack or shelve systems to save space
–At least one Rack will hold 10 PCs with typical cases

• Wheels are nice and don’t cost much more

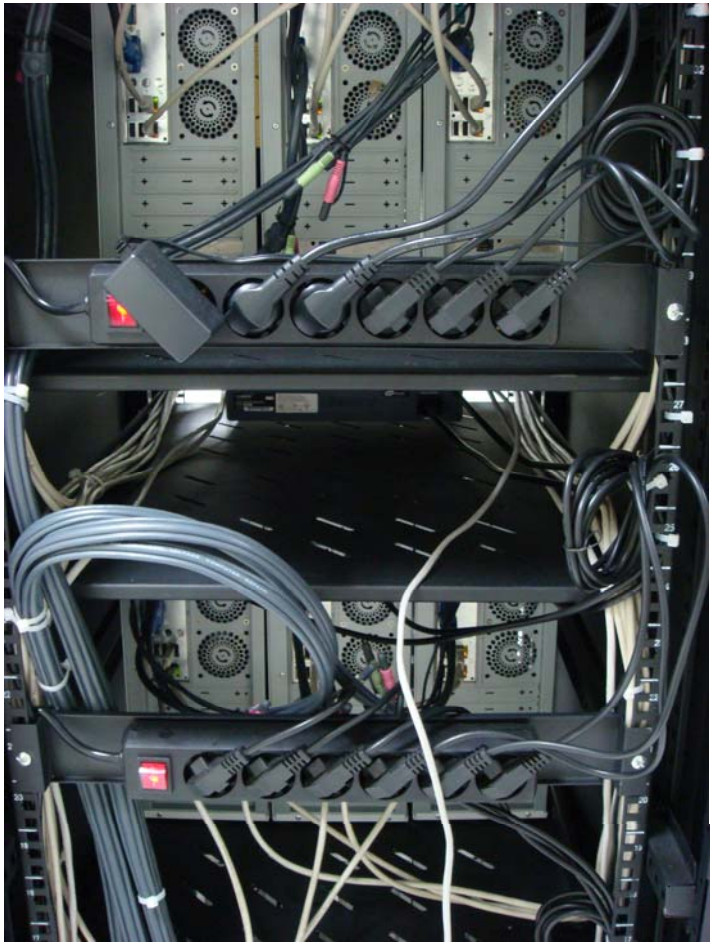
• Watch for tipping!
–Multiprocessor systems may save space
–Rack mount cases are smaller but expensive

Environment: Power

• Make sure you have enough power.
–1.3Ghz Athlon draws 1.6A at 110 Volts = 176 Watts

• Newer systems draw more; measure for yourself!
–Wall circuits typically supply about 20 Amps

• Around 12 PCs @ 176W max (8-10 for safety)



Environment:
Uninterruptable Power Systems

• 5kVA UPS ($3,000)
–Will need to work out building power to them
–Holds 24 PCs @176W (safely)
–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 –large clusters require dedicated cooling.

• Make sure you have enough cooling.
–One PC @176W puts out ~600 BTU of heat.
–1 ton of AC = 12,000 BTUs = ~3500 Watts
–Can run ~50 PCs per ton of AC (30-40 safely)

Environment: fire extinguisher

- To prepare the fire accidents.
- Automatic fire extinguisher is good.

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)

• Three main options:
–100Mbps Ethernet –very cheap ($50/node), universally supported, good for low-bandwidth requirements.
–Gigabit Ethernet –moderate ($200-300/node), well supported, fewer choices for good cards, cheap commodity switches only up to 24 ports.
–Special interconnects:

• Myrinet–very expensive ($2500/node), very low latency, logarithmic cost model for very large clusters.

Hardware: Gigabit Ethernet (1)

• The only choice for low-cost clusters up to 48 processors.

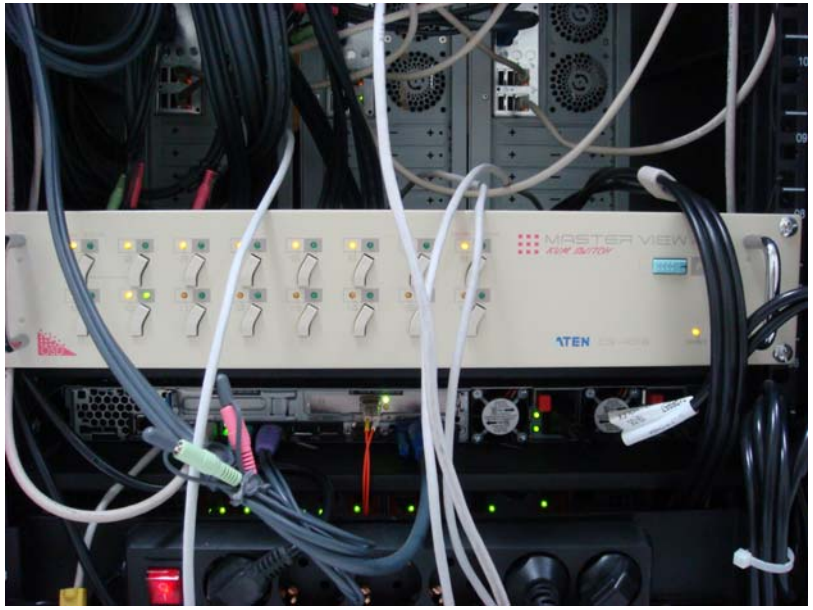
• 24-port switch allows:
–24 single nodes with 32-bit 33 MHz cards
–24 dual nodes with 64-bit 66 MHz cards

Hardware: Gigabit Ethernet (2)

• Jumbo frames:
–Extend standard ethernet maximum transmit unit (MTU) from 1500 to 9000
–More data per packet, fewer packets, lowers CPU load.
–Requires managed switch to transmit packets.
–All communicating nodes must use Jumbo frames, if enabled
–Atypical usage patterns not as well optimized.

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

• More choices: operating system, message passing libraries, numerical libraries, compilers, batch queueing, etc.

• 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)

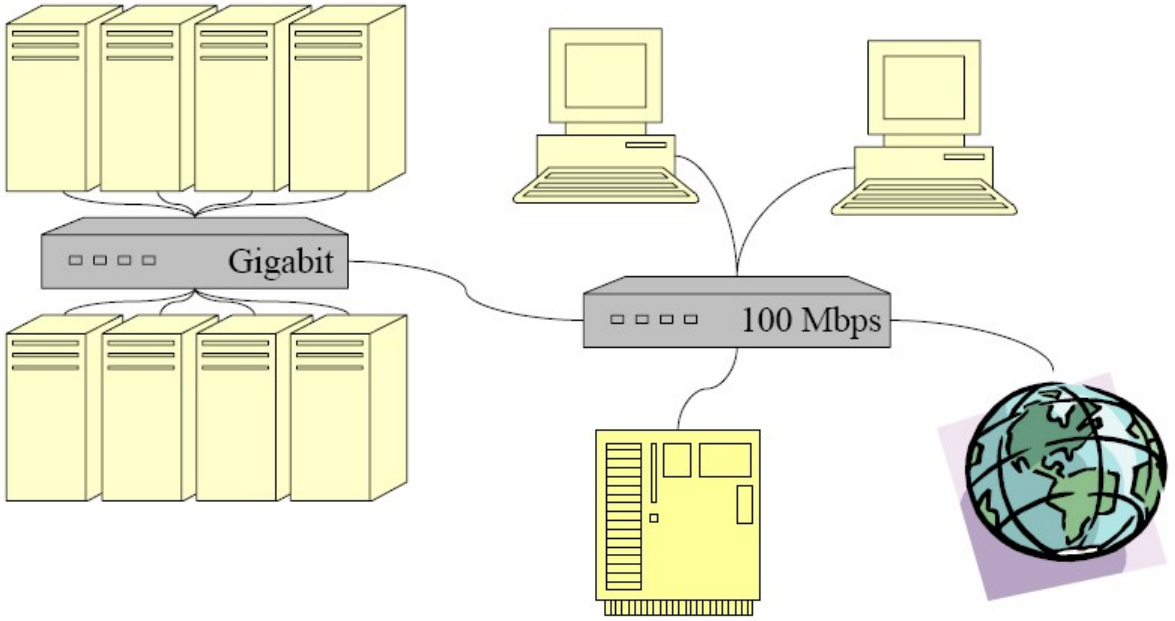
• SMP Nodes:
–Does the kernel TCP stack scale?
–Is the message passing system multithreaded?
–Does the kernel scale for system calls made by your applications?

• Network Performance:
–Optimized network drivers?
–User-space message passing?
–Eliminate unnecessary daemons, they destroy performance on large clusters (collective ops)

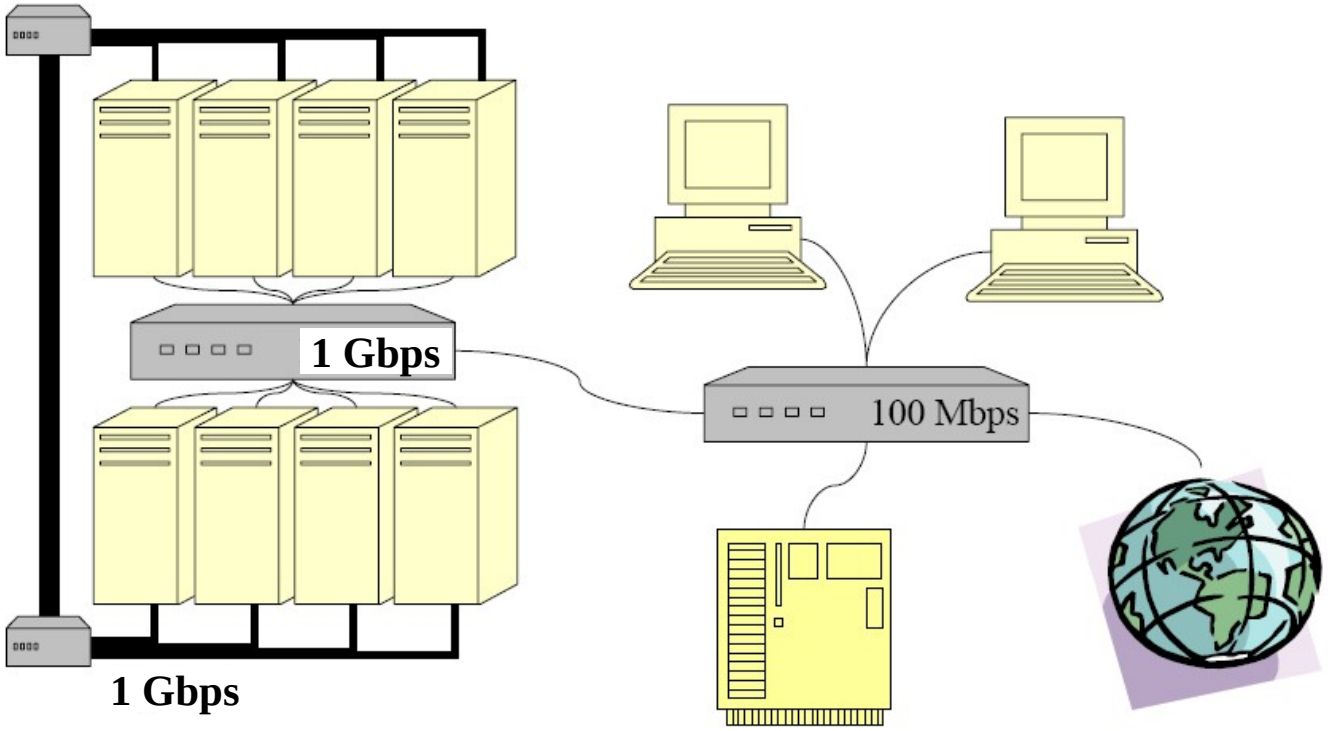
Software: Networking

• 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: APCC



Scyld Beowulf / ClusterMatic

• Single front-end master node:
–Fully operational normal Linux installation.
–Bproc kernel 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

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 evalcompiler 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
- PVM: typically low performance avoid if possible
- 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

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

Case 2005

Pentium4 PC

1. CPU : 2.0 GHz
2. Mem. : 512 MB
3. HDD : 1TB
4. NIC : 100 Mbps * 2
5. Prupose

- Data FTP
- Data Preprocessing
- Terminal server

Case 2006

Opteron PC base

1. CPU : 2.2 GHz 1way dual core * 9 nodes

2. Mem. : 1 GB (in each core)

3. HDD : 4TB

4. NIC : 1000 Mbps * 3

5. Purposes

- Operational Work

- Parallellized model operation

- Individual research works

Case 2007

Opteron PC base

1. CPU : 2.2 GHz 8way dual core * 1 node
2. Mem. : 2 GB (in each core)
3. HDD : 9TB
4. NIC : 1000 Mbps * 2, 4 Gbps SAN * 2
5. Purposes

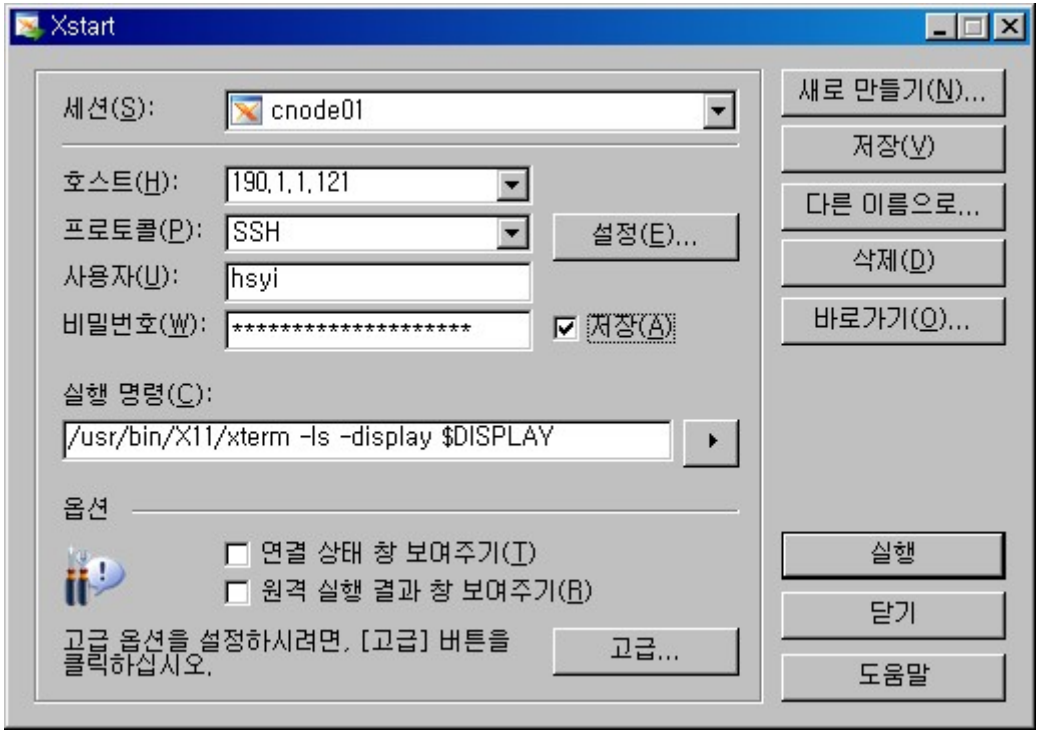
- Operational Work
- Parallellized model operation
- Individual research works

Use APCC computer

Aug. 2007

Hanse Yi

Access to clusters



Thank for your attention