# 集群技术概述

于策

# Outline

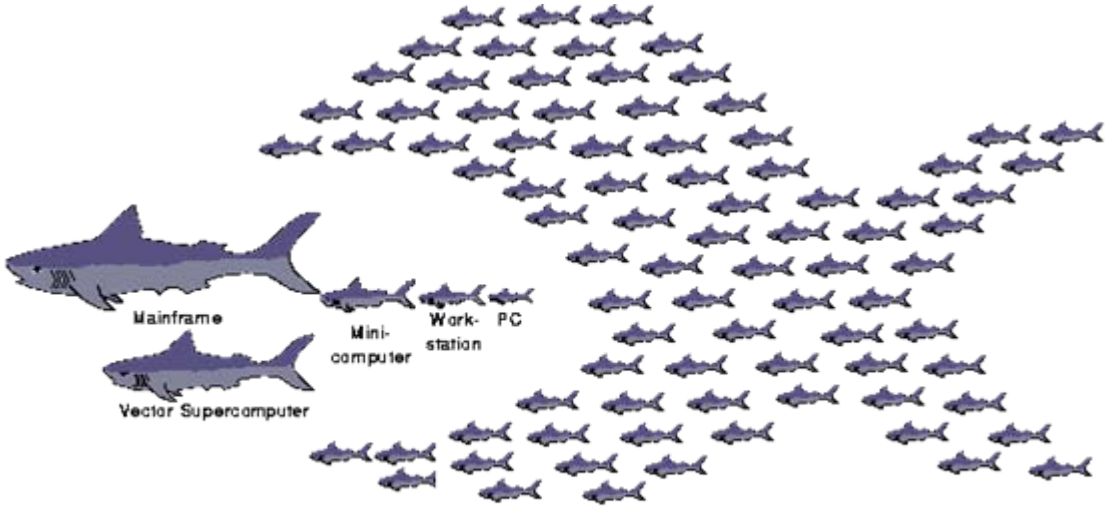
- 集群技术基础
  - 定义
  - 体系结构
  - 分类及实例
- PVM/MPI
- RSH/SSH

# Outline

- 集群技术基础
  - 定义
  - 体系结构
  - 分类及实例
- PVM/MPI
- RSH/SSH

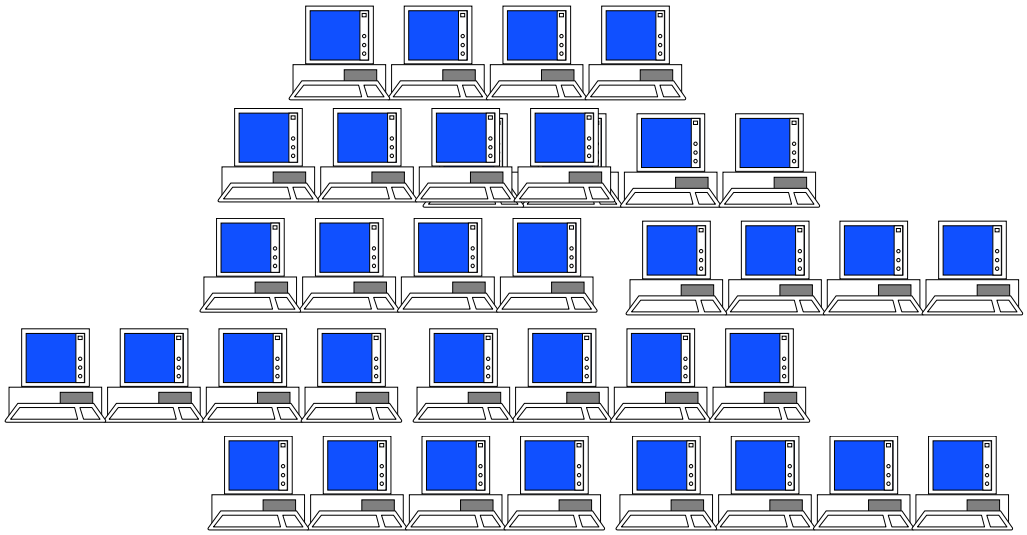
# 集群技术

- 集群概念最早由IBM于20世纪60年代提出
- 集群一般由高速网络连接起来的高性能工作站或PC机组成。集群在工作中像一个统一的整合资源，所有节点使用单一界面。




NOW

# 集群



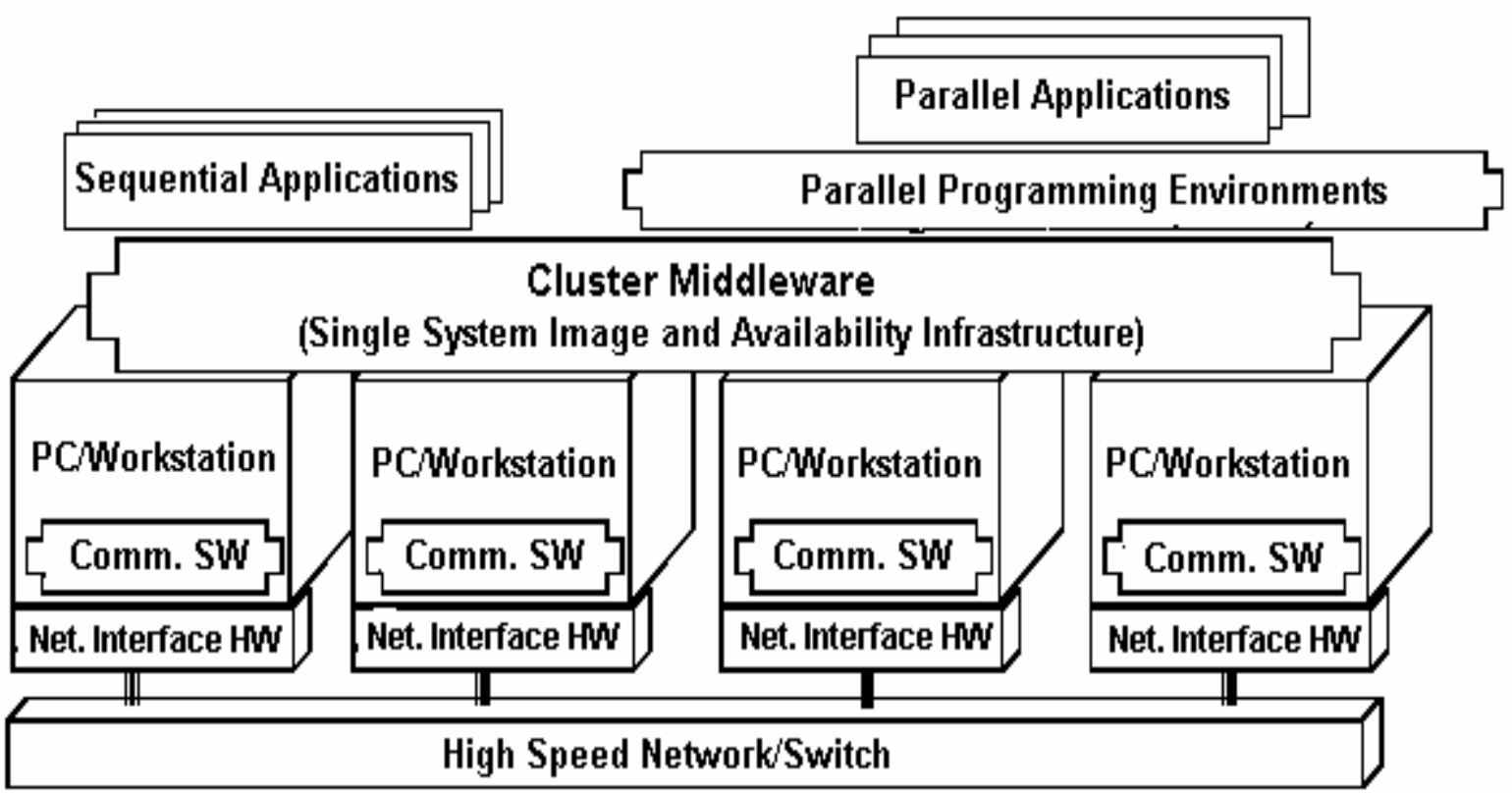
Not a Cluster

Cluster

# Outline

- 集群技术基础
  - 定义
  - 体系结构
  - 分类及实例
- PVM/MPI
- RSH/SSH

# 集群计算系统体系结构


Parallel Applications
Sequential Applications
Parallel Programming Environments
Cluster Middleware
(Single System Image and Availability Infrastructure)
PC/Workstation
Comm. SW
Net. Interface HW
PC/Workstation
Comm. SW
Net. Interface HW
PC/Workstation
Comm. SW
Net. Interface HW
PC/Workstation
Comm. SW
Net. Interface HW
High Speed Network/Switch

# Outline

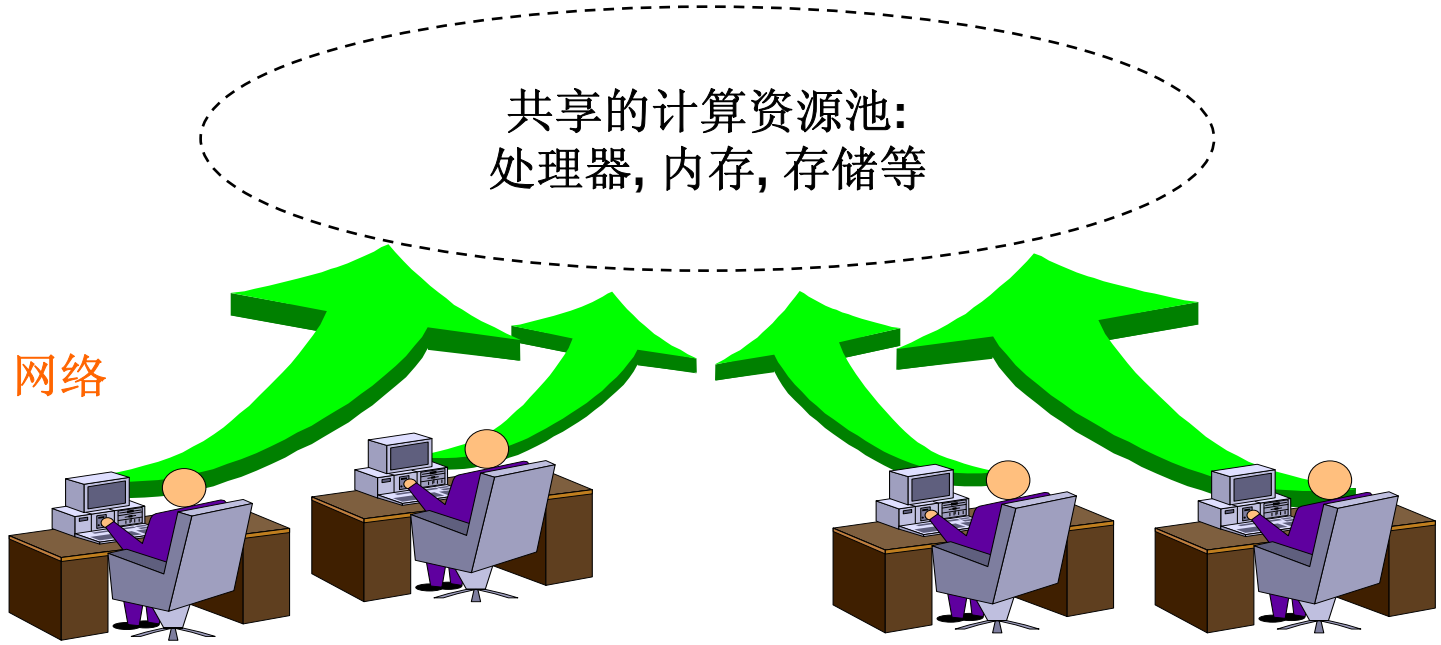
- 集群技术基础
  - 定义
  - 体系结构
  - 分类及实例
- PVM/MPI
- RSH/SSH

# 集群的分类

- 基于节点的所有者
  - 专用集群
  - 非专用集群

# 专用集群

# 非专用集群




确保每个工作人员都有至少一个节点可用，不用时收回

将集群中大部分计算资源分配给少数几个应用集中使用

# 集群的分类

- 基于节点的操作系统
  - Linux Clusters (Beowulf)
  - Solaris Clusters  (Berkeley NOW)
  - AIX Clusters (IBM SP2)
  - SCO/Compaq Clusters (Unixware)
  - Windows Clusters

# 集群的分类

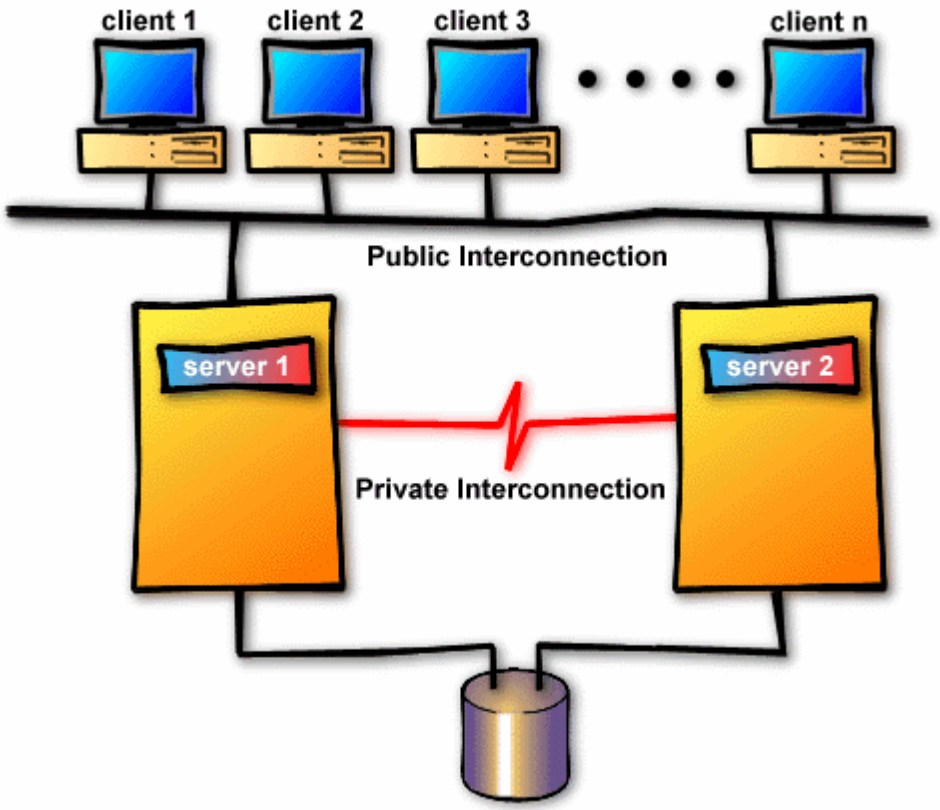
- 基于节点的体系结构、配置和操作系统等
  - 同构集群
    - 所有节点配置相同
  - 异构集群
    - 不同节点有所差异

# 集群的分类

- 用途
  - 高可用集群
    - 保证服务质量
    - 容错
    - 负载均衡
  - 高性能计算集群
    - 大规模科学计算
    - 海量数据存储与处理

# 高可用集群

- 最大程度地减少服务中断。
  - Hearbeat
  - LVS（Linux Virtual Sever）
  - IBM 的 Tivoli 和 WebSphere 系列软件
  - ……



# 高可用（HA）

- 系统可用性分类

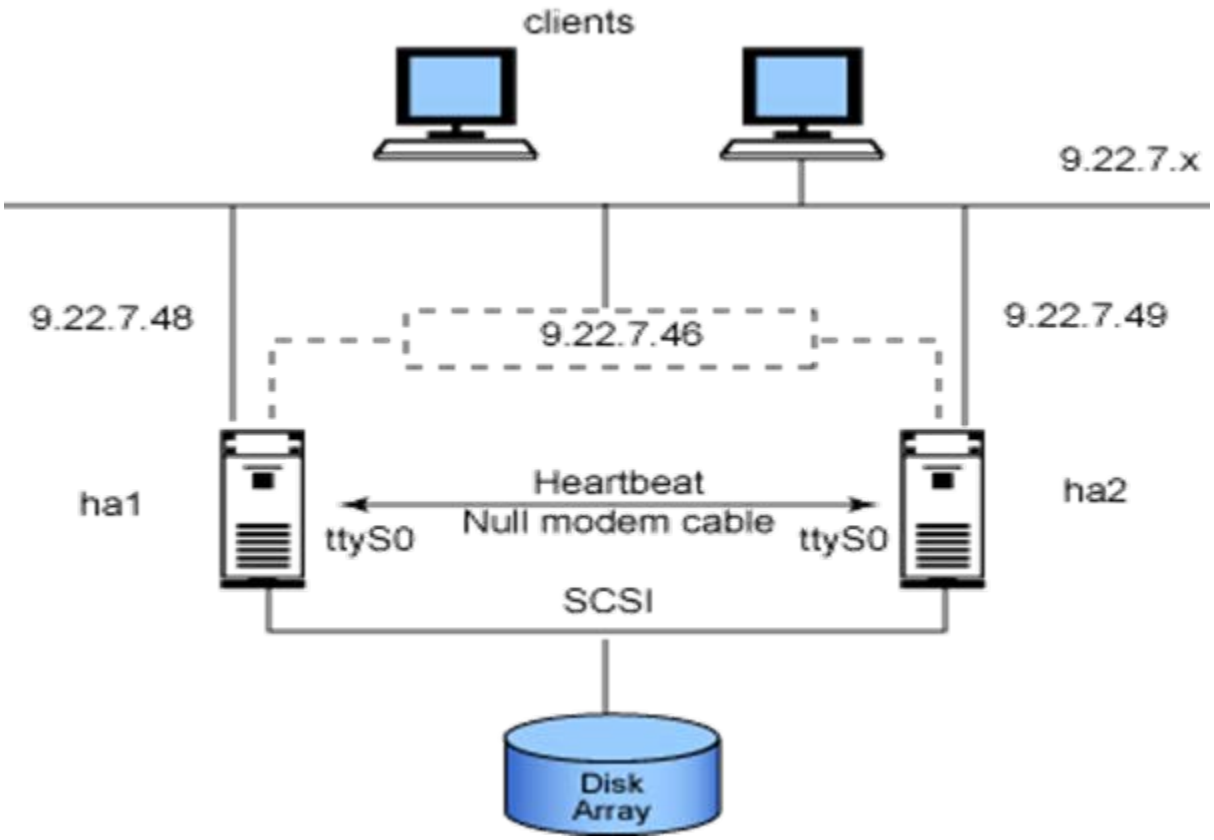
| 可用比例 (Percent Availability) | 年停机时间 (downtime/year) | 可用性分类 |
|---|---|---|
| 99.5 | 3.7天 | 常规系统(Conventional) |
| 99.9 | 8.8小时 | 可用系统(Available) |
| 99.99 | 52.6分钟 | 高可用系统(Highly Available) |
| 99.999 | 5.3分钟 | Fault Resilient |
| 99.9999 | 32秒 | Fault Tolerant |

- 停机给企业带来的损失

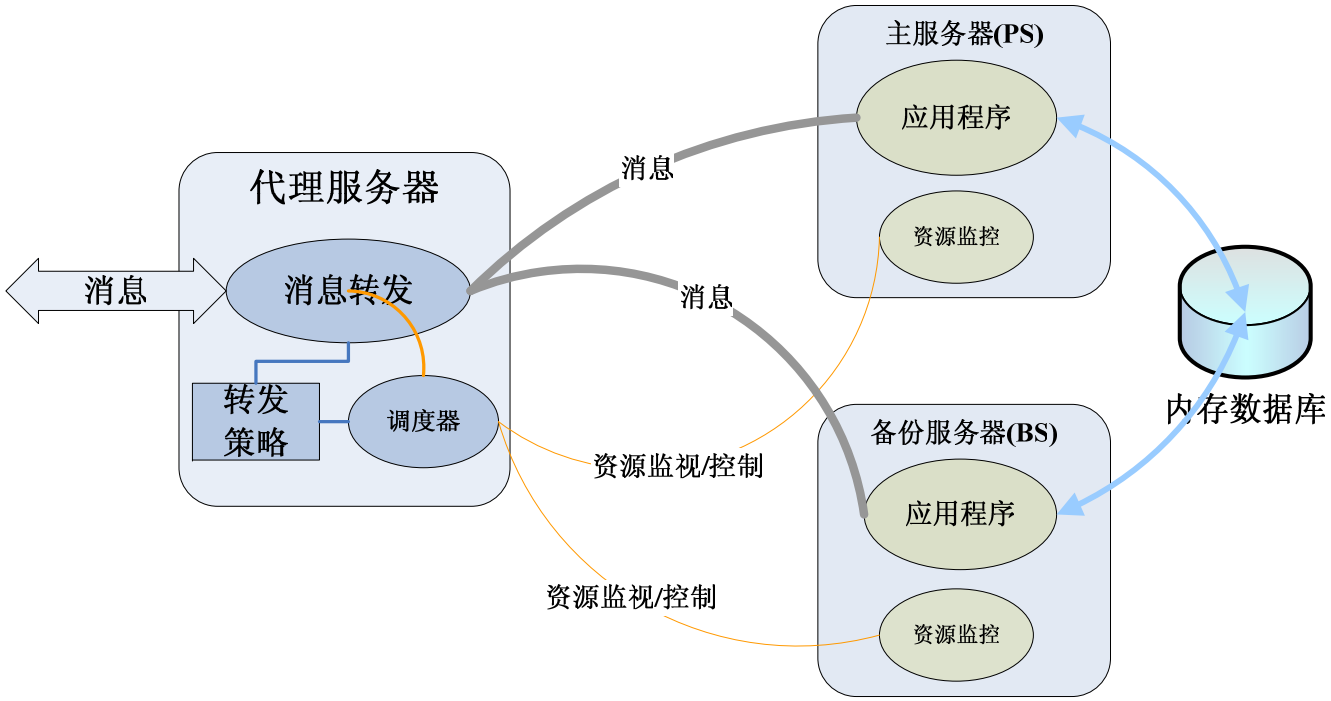
| 应用系统 | 每分钟损失(美元) |
|---|---|
| 呼叫中心(Call Center) | 27000 |
| 企业资源计划(ERP)系统 | 13000 |
| 供应链管理(SCM)系统 | 11000 |
| 电子商务(eCommerce)系统 | 10000 |
| 客户服务(Customer Service Center)系统 | 27000 |

数据来源：http://www.ibm.com/developerworks/cn/linux/cluster/hpc/part1/index.html

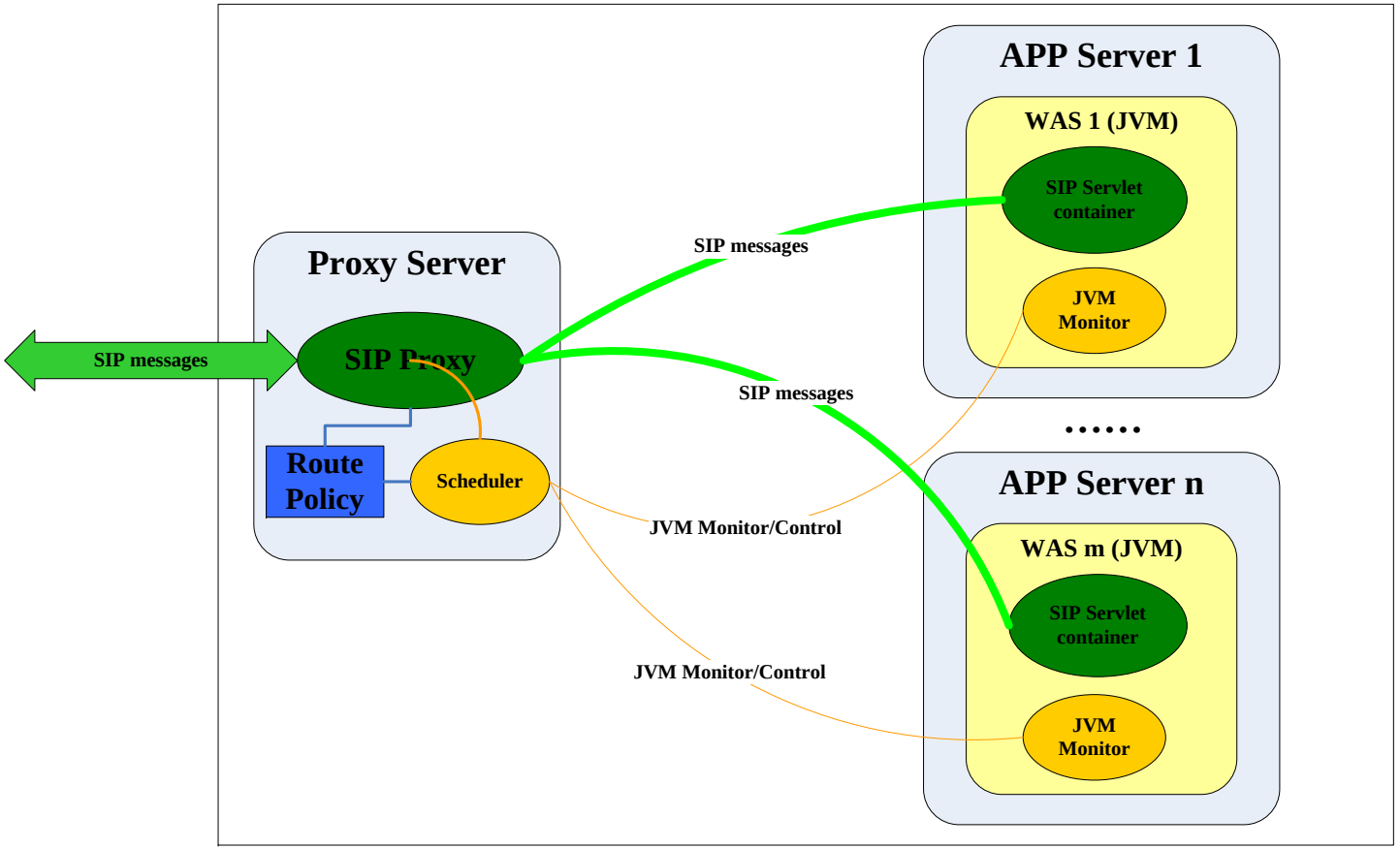
# Heartbeat 集群



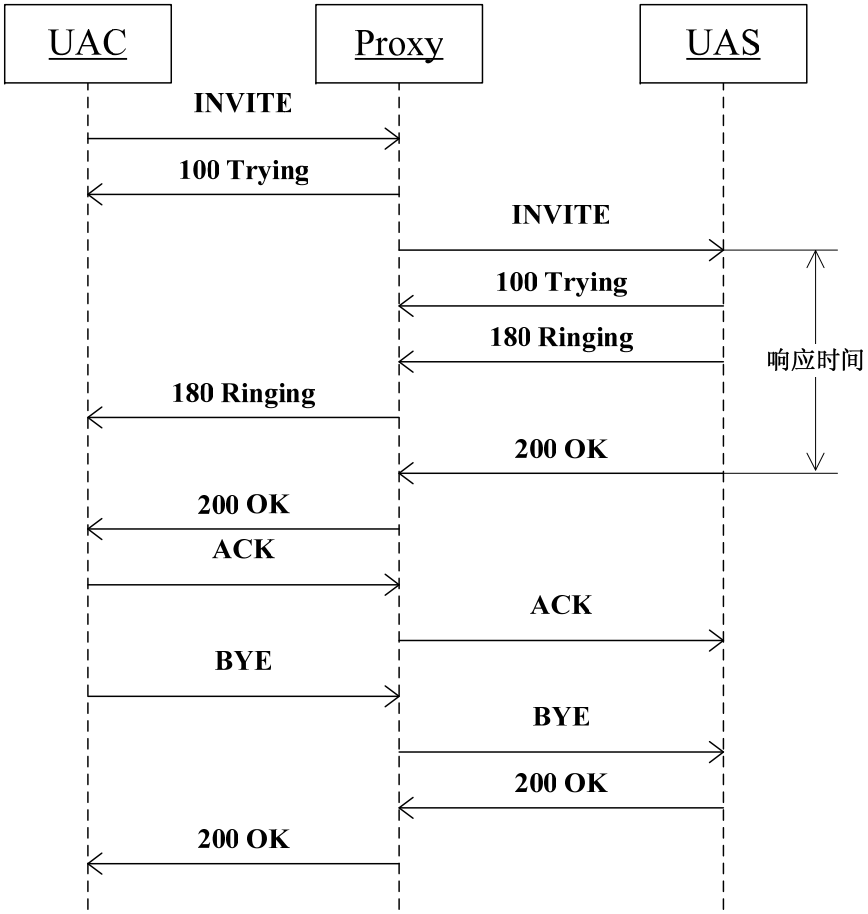
# 高可用集群实例


主服务器(PS)
应用程序
资源监控
代理服务器
消息
消息转发
转发策略
调度器
消息
消息
资源监视/控制
资源监视/控制
备份服务器(BS)
应用程序
资源监控
内存数据库

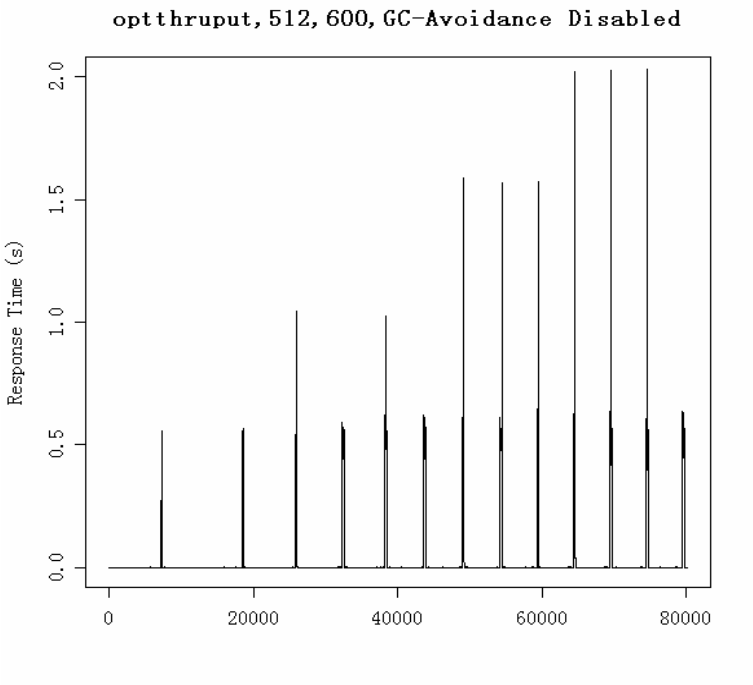
# 实验配置

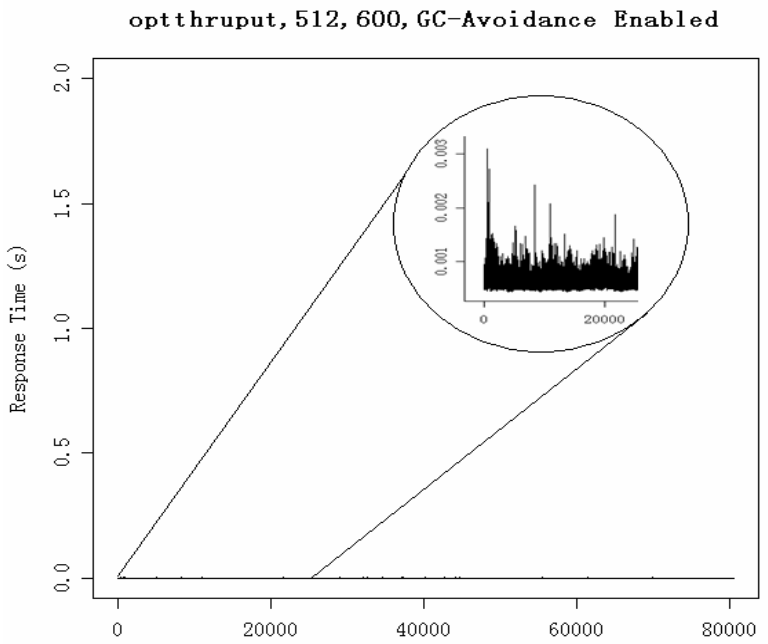

APP Server 1
WAS 1 (JVM)
SIP Servlet container
JVM Monitor
Proxy Server
SIP messages
SIP Proxy
Route Policy
Scheduler
SIP messages
SIP messages
JVM Monitor/Control
JVM Monitor/Control
APP Server n
WAS m (JVM)
SIP Servlet container
JVM Monitor

# SIP服务


UAC
Proxy
UAS
INVITE
100 Trying
INVITE
100 Trying
180 Ringing
响应时间
180 Ringing
200 OK
200 OK
ACK
ACK
BYE
BYE
200 OK
200 OK

# 实验结果




专利：

- ✓ “用于系列服务消息处理的方法、设备和系统”，中国专利， CN101242392，2008.8.13
- ✓ “METHOD, APPARATUS AND SYSTEM FOR PROCESSING A SERIES OF SERVICE MESSAGES”, US Patent, US20080195718, 08/14/2008

# 负载均衡集群

- Oracle数据库集群
- IBM服务器集群

# 高性能计算集群

- Cluster1350
- 天津大学超算中心曙光集群

# Beowulf 集群

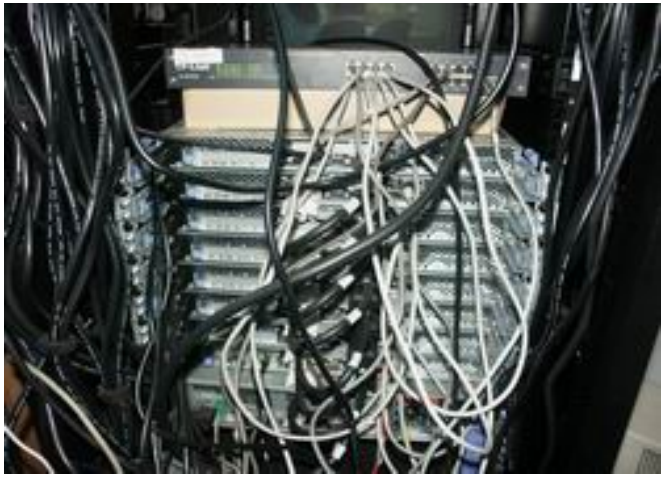
- 标准的、商品化的、廉价的高性能处理器
- 高速网络技术
- 免费、开放的系统及并行软件
- http://www.beowulf.org/

# IBM Cluster1350

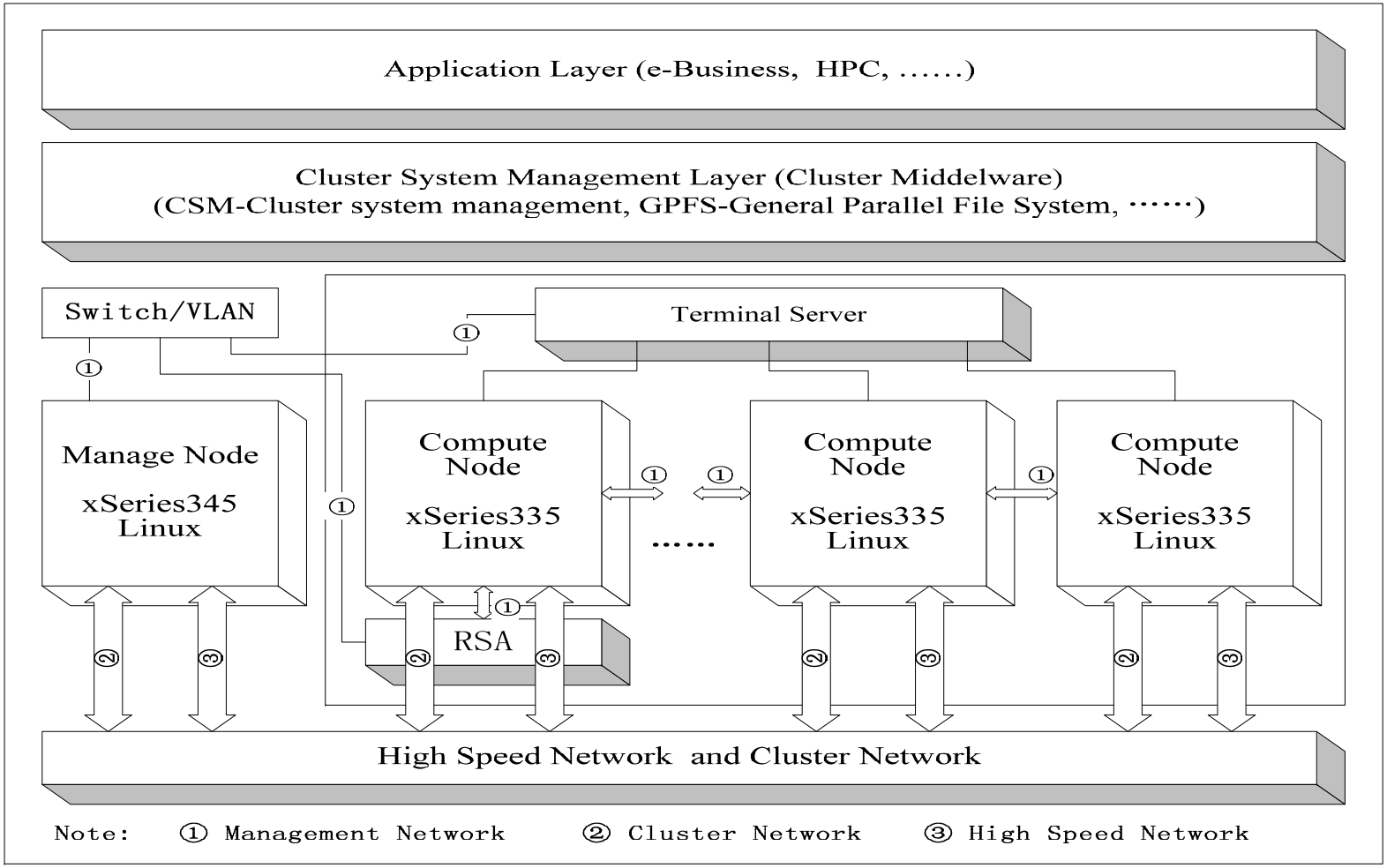
- Cluster1350是IBM公司目标定位于高性能计算市场的Linux集群，包括一套完整的解决方案，集成了众多IBM与非IBM的先进的软硬件技术，有其特有的技术优势与强大的服务支持。

http://www.ibm.com/developerworks/cn/linux/cluster/l-ibm1350/index.html

# Cluster1350



# Cluster1350逻辑结构



# Cluster1350 @ TJU

- 管理节点
  - IBM eServer xSeries 345 (1)
- 计算节点
  - IBM eServer xSeries 335 (16)
- 网络
  - 千兆以太网
- 操作系统
  - Red Hat Linux 7.3

# High Speed Network

- Cluster1350的计算网络可选Myrinet超高速网络或者千兆以太网，以及相应的通信协议，用于并行计算时各结点间数据交换。
  - 实际使用的是千兆以太网

# Manage Node

- Cluster1350的管理节点为xSeries345 (2U)，操作系统为Linux，目前支持RedHat 7.2与7.3，RedHat AS2.1，以及SuSe 8.0和8.1，SuSe SLES7.2和8.0。自带两个10M/100M/1000M自适应网卡，支持RAID，有RSA适配器接口(PCI插槽)。

# xSeries345


*Figure 2-2 Model 345 for storage or management nodes*

# Compute Node

- Cluster1350的计算结点为xSeries335 (1U)，操作系统为Linux，目前支持RedHat 7.3，RedHat AS2.1，以及SuSe 8.0和8.1，SuSe SLES7.2和8.0。自带两个10M/100M/1000M自适应网卡，有RSA适配器接口(PCI插槽)。

# xSeries335

*Figure 2-3 Model 335 for cluster (compute) nodes*

# Terminal Server

- 各结点通过串口连接到Terminal Server，通过Terminal Server，管理员在管理结点上可以获得任意受控结点的控制台，而不管该结点在普通网络(Management Network)上是否可达。一个Cluster1350集群根据规模不同，可以有一个或多个Terminal Server。在结点比较少时，也可以不用Terminal Server，而用KVM交换机以及xSeries335前面板上的控制按钮配合来实现控制台切换，不过后一种方式当结点数目增多时连接及操作复杂度会越来越高。

# RSA (Remote Supervisor Adapter)

- RSA适配器结点机主板上的ISMP以及C2T Chain等其它相关硬件配合工作，用于实现对集群中各结点的电源管理、机器硬件状态监测、日志报告等管理功能，是Cluster1350中硬件控制的接入点。一个Cluster1350集群中可以有多个RSA配置器，每一个RSA适配器最多可控制24个结点。

# Management Network

- Cluser1350的集群管理网络由各结点上的ISMP (Integrated Systems Management processor)、C2T Chain (Cable Chain Technology)、RSA适配器、Terminal Server、Management Switch/VLAN构成。其中ISMP内置于安结点主板，由C2T Chain级联，然后通过RSA适配器用网线连接到管理网络；各结点用串口线连接到Terminal Server，Terminal Server也通过网线连接到管理网络。这样，管理结点通过管理网络可以便捷地实现对集群所有结点的控制。

# Management Network

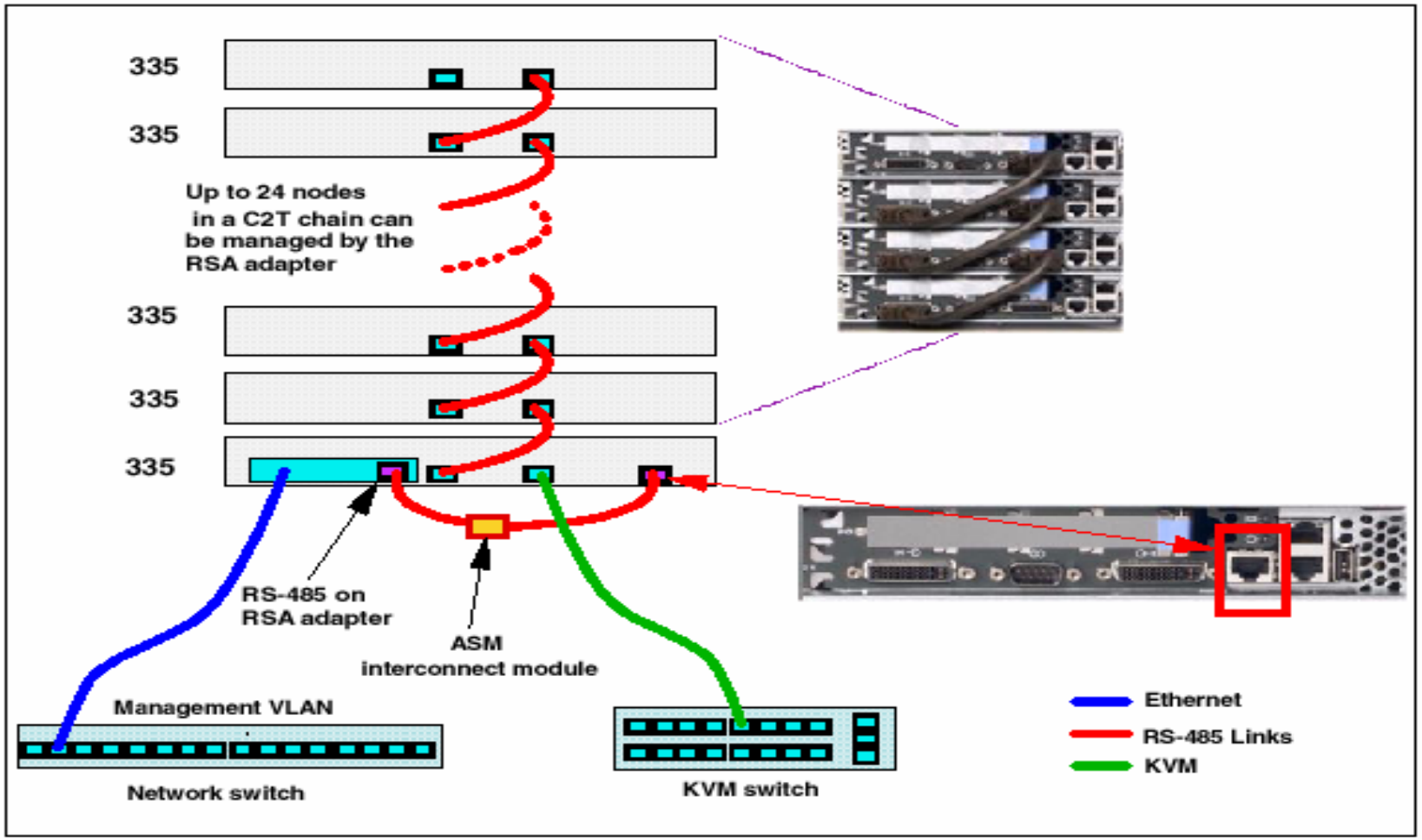



*Figure 2-4 Management processor network*

# Cluster Network

- Cluster Network可以是普通的网络，主要用于集群系统管理软件对集群的管理，比如监控结点状态、网络安装各结点操作系统、更新各结点配置文件及软件等。Cluster Network一般不用于并行计算时各结点间数据交换。

# Cluster1350网络互联

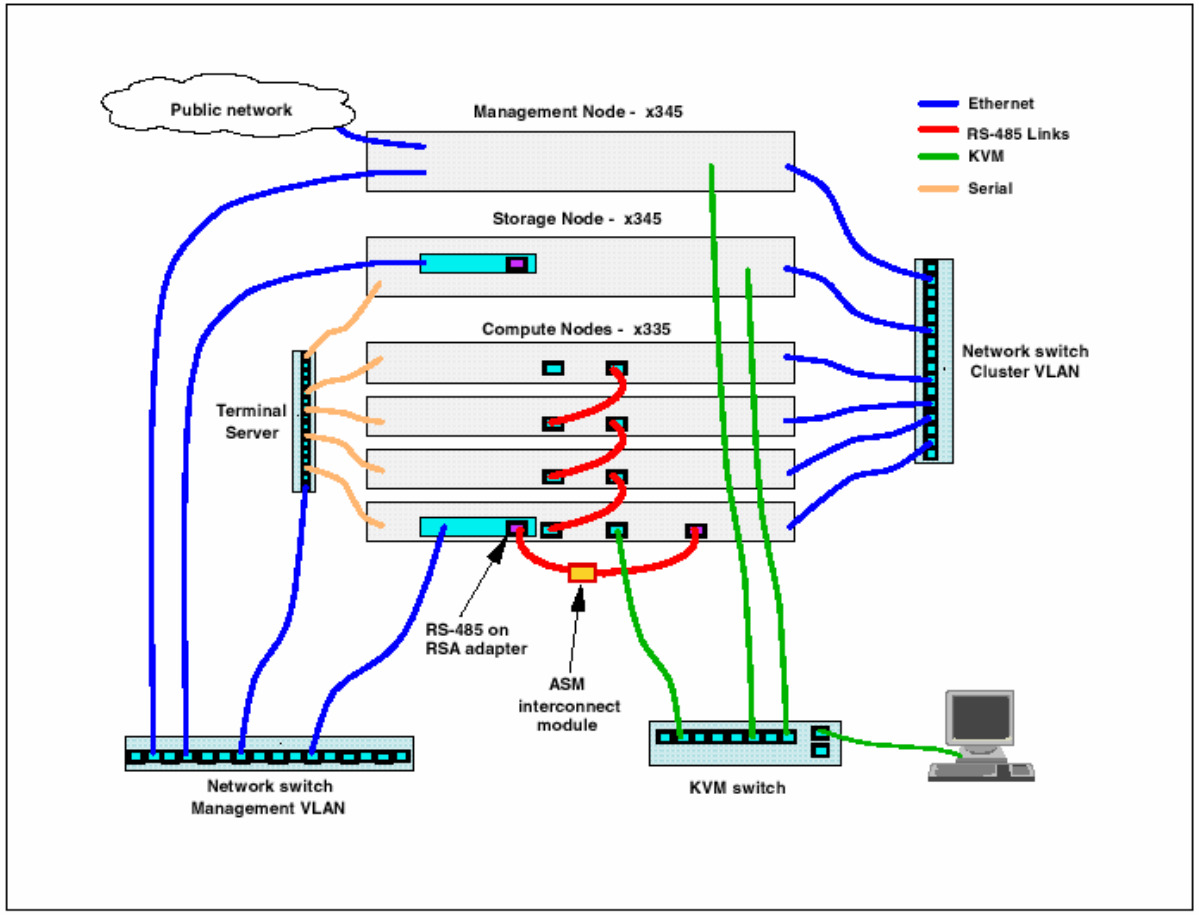
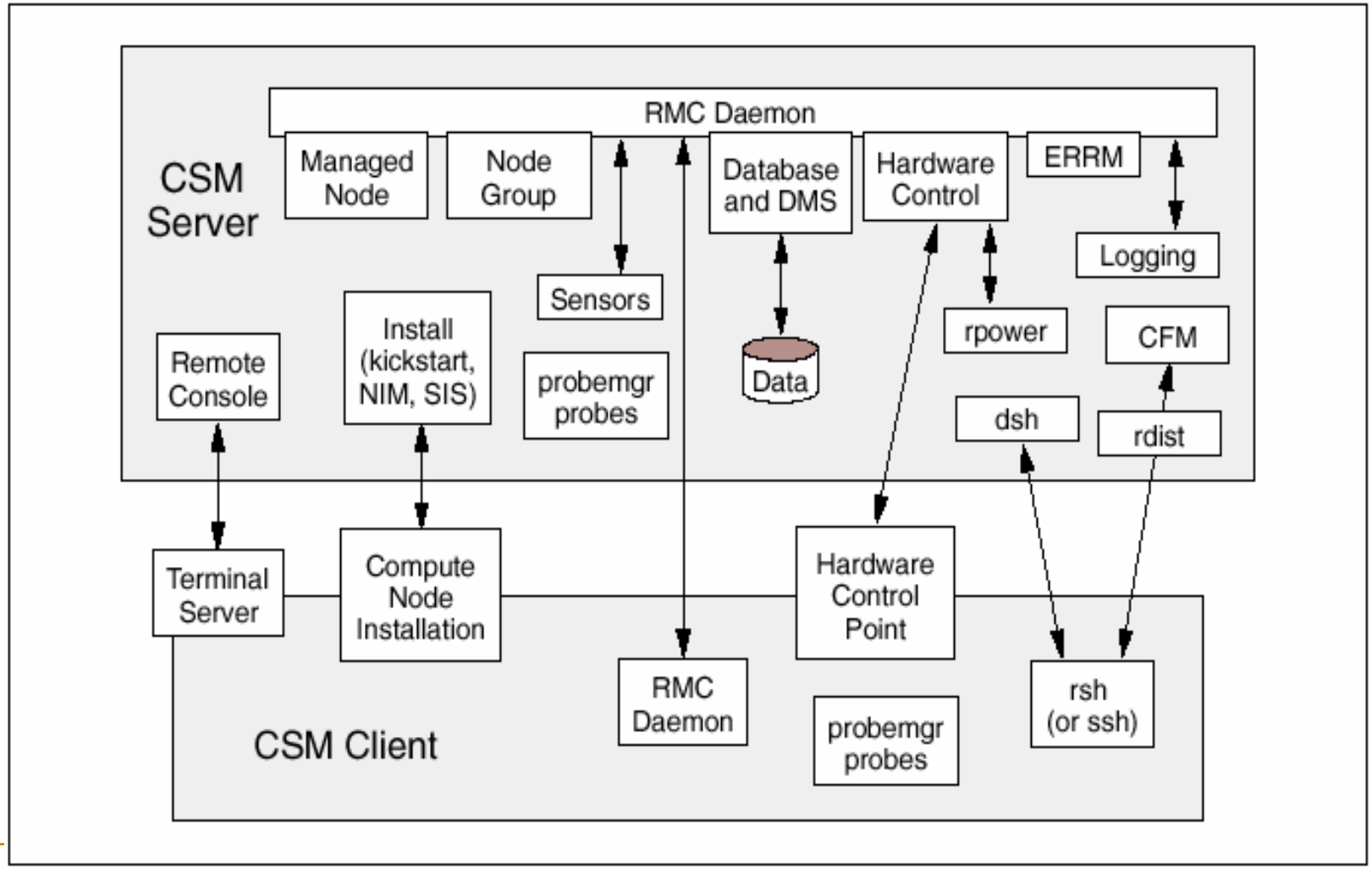



*Figure 5-1 Lab cluster configuration*

# CSM（Cluster System anagement）

- CSM是IBM公司开发，专门用于集群系统管理的中间件，在Cluster1350解决方案集成。
  - CSM的设计思想与体系结构来自PSSP (IBM Parallel System Support Programs for AIX)与其它一些开源的集群管理软件。还有一些中间件及技术，虽然不直接为用户服务，但构成了CSM的不可或缺的基础，包括RMC、SRC、RSCT等。

# CSM体系结构

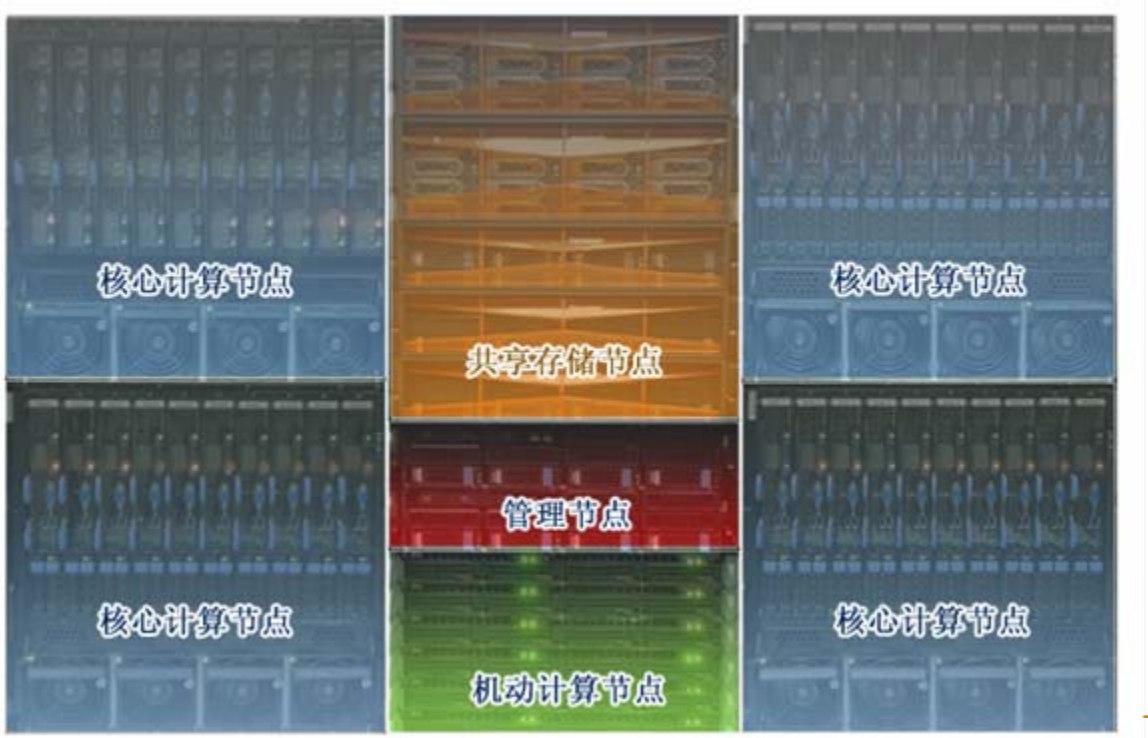

RMC Daemon
CSM Server
Managed Node
Node Group
Database and DMS
Hardware Control
ERRM
Logging
Sensors
Install (kickstart, NIM, SIS)
Remote Console
probemgr probes
Data
rpower
CFM
dsh
rdist
Terminal Server
Compute Node Installation
Hardware Control Point
RMC Daemon
probemgr probes
rsh (or ssh)
CSM Client

# Cluster1350系统管理

- 整个集群由单一结点控制
  - 所有结点的
    - 开机、关机、状态查询
    - 显示远程控制台
    - 安装操作系统
    - 升级(安装)各结点系统及应用软件
    - 。。。
  - 一个完整的集群只需一套外置输入/输出设备(键盘、鼠标、显示器)

# 天津大学超算中心

- 曙光“星云”系列高性能计算集群包含56个计算节点，1232个CPU核，内存总容量2TB，并行存储总容量33TB，峰值速度可以达到每秒11万亿次。



# 管理节点

- 用户登录、软件安装，程序编译、任务提交、作业管理和系统监控

| 管理节点 Node61/62 | |
|---|---|
| 处理器 | 2 块 4 核 AMD Opteron Processor 4130 CPU，主频 2.6GHz |
| 内存 | 16GB DDR3 1333MHz ECC |
| 网络连接 | 双 1000M 以太网接口<br>可支持网卡冗余、网络唤醒、负载均衡； |
| 操作系统 | SUSE Linux Enterprise Server 10 SP2 |

# 核心计算节点

| 计算节点 Node 1 - Node 30 | |
|---|---|
| 处理器 | 2 块 12 核 AMD Opteron Processor 6174 CPU，主频 2.2GHz |
| 内存 | 48GB DDR3 1333MHz ECC |
| 网络连接 | 2 块 1000M 以太网接口<br>1 个 40Gb Infiniband 接口 |
| 操作系统 | SUSE Linux Enterprise Server 10 SP2 |

# 核心计算节点

| 计算节点 Node 31 - Node 40 | |
|---|---|
| 处理器 | 4 块 8 核 AMD Opteron Processor 6132 HE CPU，主频 2.2GHz |
| 内存 | 64GB DDR3 1333MHz ECC |
| 网络连接 | 2 块 1000M 以太网接口<br>1 个 40Gb Infiniband 接口 |
| 操作系统 | SUSE Linux Enterprise Server 10 SP2 |

# 机动计算节点

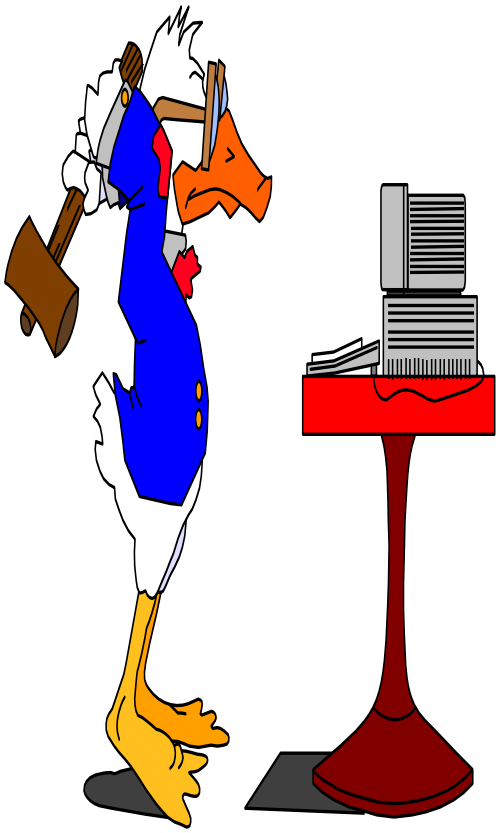
- 用户试用、操作培训、软件测试与教学实验需要，在核心计算节点负荷过大、设备故障或系统维护期间，作为机动计算资源试用

| 计算节点 Node 41 - Node 56 | |
|---|---|
| 处理器 | 2 块 6 核 AMD Opteron Processor 2435 CPU，主频 2.4GHz |
| 内存 | 16GB DDR3 1333MHz ECC |
| 网络连接 | 2 块 1000M 以太网接口<br>1 个 20Gb Infiniband 接口 |
| 操作系统 | SUSE Linux Enterprise Server 11 |

# 共享存储节点

- 计算集群用户的数据均存放在共享存储节点上，存储节点挂载在各个计算节点下，确保用户在所用节点均可访问存储上的数据，用户不能直接登录存储节点。并行存储系统提供海量存储空间，总共配置33TB SATA介质智能磁盘。

# 集群设计中需要考虑的问题



- 单一系统映像 (对外如同单一系统)
- 高速通信 (网络 & 协议)
- 负载均衡 (任务队列与调度)
- 存储 (并行文件系统)
- 可编程性 (API，中间件)
- 集群规模的扩展性 (硬件& 应用)
- 高可用性 (容错)
- 适用性 (可以支持的应用类型)
- 安全与加密

# 集群上的应用程序（计算）开发

- 适于开发基于消息传递的并行应用程序
  - 可以使用PVM/MPI
- 步骤：
  - 配置并行计算的编译与运行环境
    - 主要是rsh/ssh
  - 设计、编写、编译程序
    - 使用PVM/MPI提供的程序库及编译环境
  - 部署应用程序
    - ftp、rcp、scp、CSM、NFS、GPFS等方式
  - 运行程序

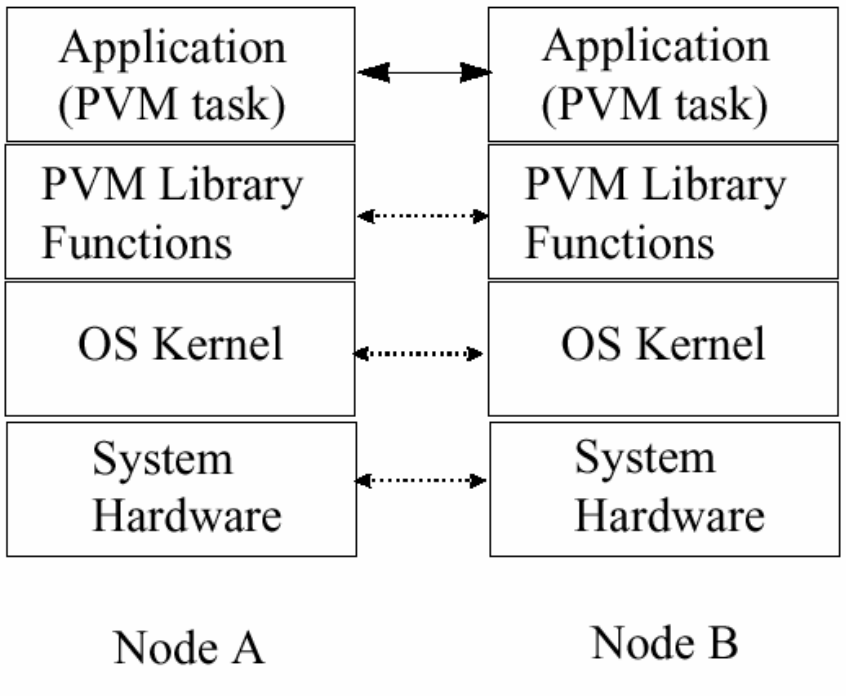
# Outline

- 集群技术基础
  - 定义
  - 体系结构
  - 分类及实例
- **PVM/MPI**
- RSH/SSH

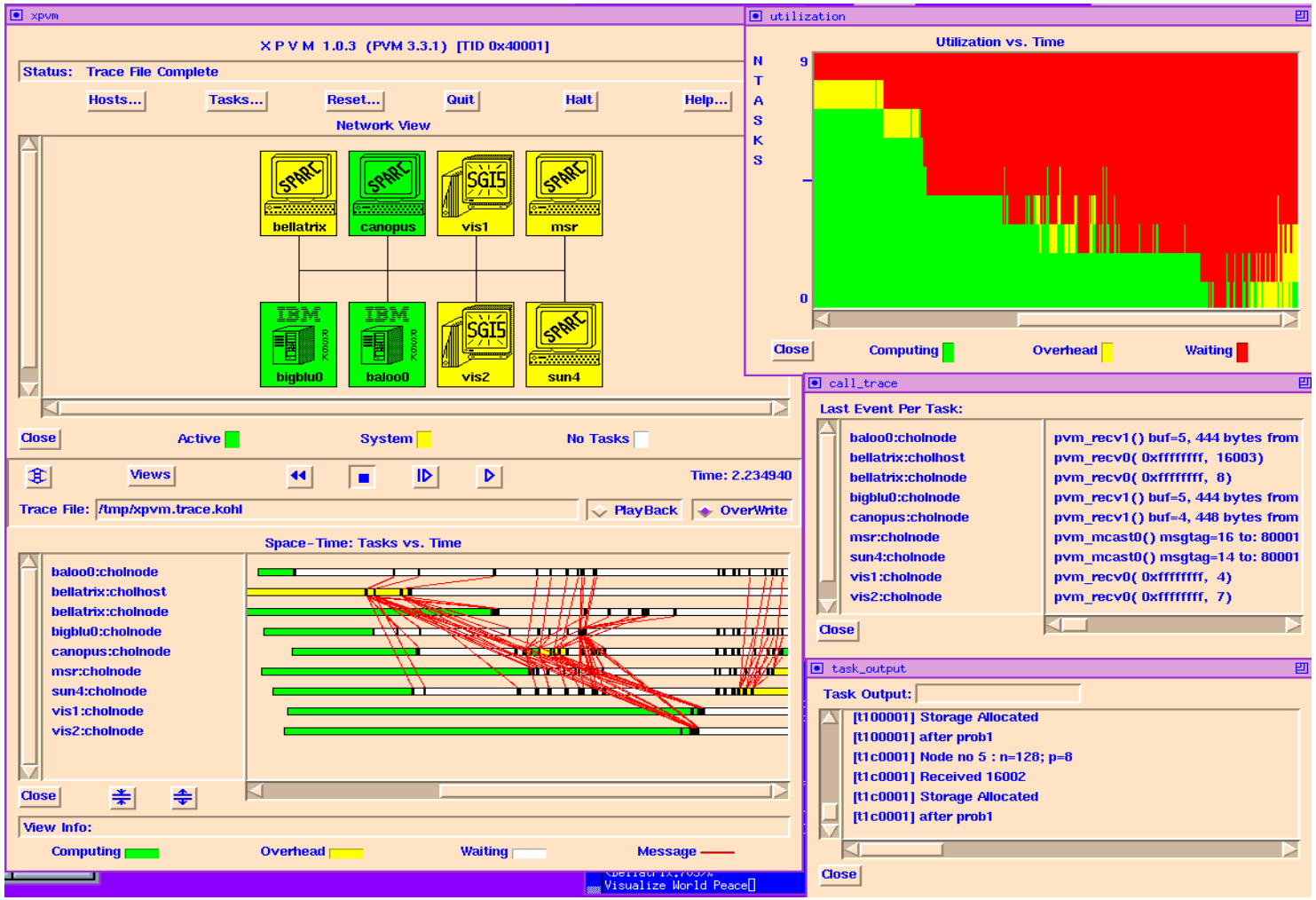
# PVM: Parallel Virtual Machine

- PVM是用于网络并行计算机上的软件工具。设计它的目的是将异构的计算机网络连接起来，使它使用起来就像一组分布式的并行处理器。
- PVM最早由美国的田纳西大学，橡树岭国家实验室以及埃默里大学开发成功。第一个版本在ORNL（橡树岭国家实验室）于1989年写成，后来，田纳西大学将其重写，并于1991年发布了版本2。版本3于1993年发布，支持容错以及更好的可移动性。

# PVM计算模型


Application (PVM task)
Application (PVM task)
PVM Library Functions
PVM Library Functions
OS Kernel
OS Kernel
System Hardware
System Hardware
Node A
Node B

# XPVM


xpvm
XPVM 1.0.3 (PVM 3.3.1) [TID 0x40001]
Status: Trace File Complete
Hosts... Tasks... Reset... Quit Halt Help...
Network View
bellatrix
canopus
vis1
msr
bigblu0
baloo0
vis2
sun4
Close
Active
System
No Tasks
Views
Time: 2.234940
Trace File: /tmp/xpvm.trace.kohl
PlayBack
OverWrite
Space-Time: Tasks vs. Time
baloo0:cholnode
bellatrix:cholhost
bellatrix:cholnode
bigblu0:cholnode
canopus:cholnode
msr:cholnode
sun4:cholnode
vis1:cholnode
vis2:cholnode
Close
View Info:
Computing
Overhead
Waiting
Message
Visualize World Peace[]
utilization
Utilization vs. Time
NTASKS
9
0
Close
Computing
Overhead
Waiting
call_trace
Last Event Per Task:
baloo0:cholnode pvm_recv1() buf=5, 444 bytes from
bellatrix:cholhost pvm_recv0( 0xffffffff, 16003)
bellatrix:cholnode pvm_recv0( 0xffffffff, 8)
bigblu0:cholnode pvm_recv1() buf=5, 444 bytes from
canopus:cholnode pvm_recv1() buf=4, 448 bytes from
msr:cholnode pvm_mcast0() msgtag=16 to: 80001
sun4:cholnode pvm_mcast0() msgtag=14 to: 80001
vis1:cholnode pvm_recv0( 0xffffffff, 4)
vis2:cholnode pvm_recv0( 0xffffffff, 7)
Close
task_output
Task Output:
[t100001] Storage Allocated
[t100001] after prob1
[t1c0001] Node no 5 : n=128; p=8
[t1c0001] Received 16002
[t1c0001] Storage Allocated
[t1c0001] after prob1
Close

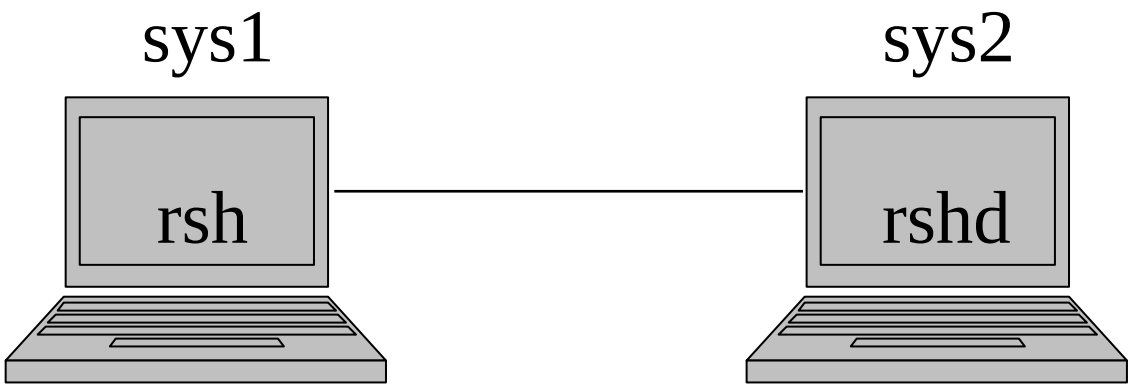
# MPI: Message Passing Interface

- MPI是一种消息传递编程模型，并成为这种编程模型的代表。
- MPI是一种标准或规范的代表，而不特指某一个对它的具体实现。
- MPI的具体实现是一个程序库，而不是一门语言。

# Outline

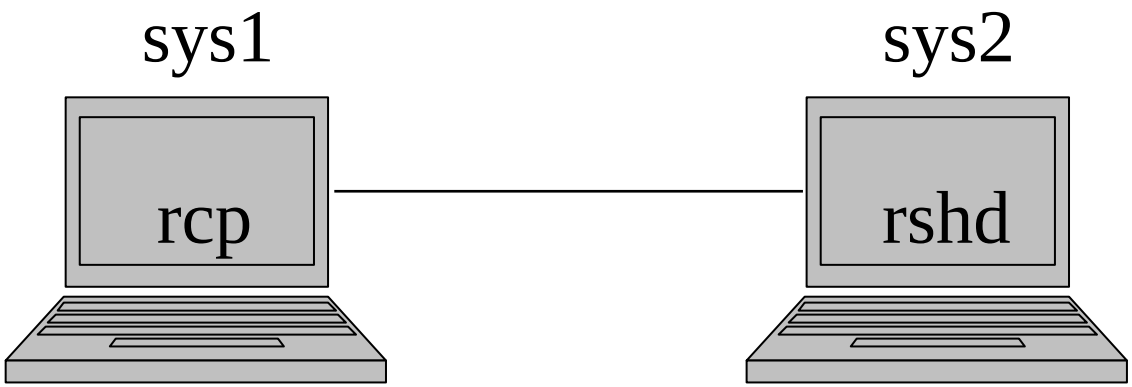
- 集群技术基础
  - 定义
  - 体系结构
  - 分类及实例
- PVM/MPI
- **RSH/SSH**

# rsh




```
sys1>whoami
        team02
sys1>rsh sys2 date
sys1>rsh sys2 –l tem04 date
sys1>rsh sys2
```

# rcp




```
sys1>whoami
        team02
sys1>rcp filea sys2:fileb
sys1>rcp filea team04@sys2:fileb
sys1>rcp –p –r sys2:dir sys4:dir
```

# 配置rsh环境

- 在各个节点的用户根目录下创建.rhosts文件，内容如下（只是类似，假定用户user01)：
  - managenode user01

    node01 user01

    node02 user01
- 将文件属性值改为600
  - chmod 600 .rhosts
- 检验rsh是否正确配置的方法（以node01为例）
  - 在managenode上，执行

    rsh node01 date

# ssh

- SSH的s命令是为了用来替代r 命令而设计的。设计者让s 命令的使用和命名与r 命令一致，以便更容易掌握。当SSH 正确安装和配置以后,s 命令提供了对用户透明的安全特性。
- 与伯克利版本提供的服务不同，SSH 的命令仅仅使用了一个守护进程sshd 和一个TCP端口,由于只用了一个进程来管理服务，使得SSH 易于监控和配置

# ssh

- SSH 提供的客户端命令
  - ssh 安全远程shell
  - slogin 安全远程登录
  - scp 安全远程拷贝
  - sftp 安全文件传输(只有SSH2 提供了sftp客户端。)
- 系统已不易受到IP 欺骗、IP源路径(IP source routing)或DNS 欺骗技术的攻击。因为这些网络攻击中可以直接看到或改变的包已经被加密了。不仅数据本身，而且那些包信息(序列号和其他一些关键性信息)都得到了保护。

# ssh工作原理

SSH有两部分：客户端和服务器程序。服务器程序是一个守护进程，它在后台运行且无须任何类型的常规管理，并响应来自客户端的连接请求。客户端提供了用户界面。

服务器端包含一个文件,即sshd程序。它通常被放在目录/usr/local/sbin下。服务器端提供了对远程连接的处理，包括公共密钥认证、密钥交换、对称密钥加密和非安全连接本身。对SSH2 来讲,用sftp-server 来管理安全文件传输的连接。

客户端包括几个不同的文件。这些文件包括ssh (该文件允许不用登录就可以在一个远程机器上运行程序)、远程拷贝( scp )、远程登录( slogin )。SSH2 有一个安全文件传输客户端( sftp )，它使用安全文件传输来替代文件传输协议(File Transfer Protocol ， FTP )。因为
FTP 不安全，所以SSH 使用自己的客户端替代它。

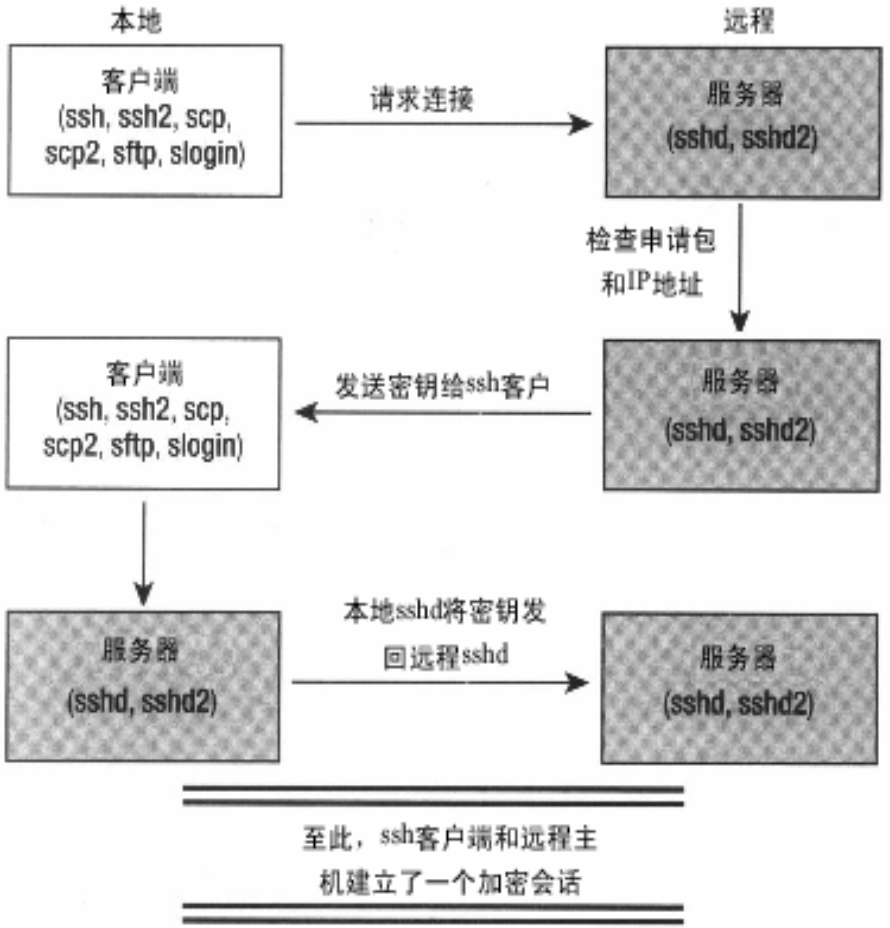



图1-2 SSH服务器和客户端的工作机制

# SSH配置步骤

- 假设所使用节点为Managenode、node01、node02。
- 方法：
  - 登录到 Managenode，输入命令：

    ssh-keygen -t rsa
  - 输入三次回车
  - 这将在用户的根目录下生成一个 .ssh 的目录，里面有三个文件。

# SSH配置步骤

- 在node01和node02 对应的用户根目录下创建 .ssh 目录
- 将Managenode节点上用户根目录下 .ssh 目录中的 id_rsa.pub 文件拷贝到上一步骤创建的目录下，并更名为 authorized_keys
- 注：第一次进行 ssh 登录时会系统会有一次询问，此时要输入“yes”，而不能输入“y”。