Linux高性能计算集群配置

李会民
hmli@ustc.edu.cn

中国科学技术大学 超级计算中心

2018-8-20

内容

高性能计算、超级计算、并行计算

- 高性能计算(HPC) 指通常使用很多处理器（作为单个机器的一部分）或者某一集群中组织的几台计算机（作为单个计算资源操作）的计算系统和环境。有许多类型的HPC系统，其范围从标准计算机的大型集群，到高度专用的硬件。大多数基于集群的HPC系统使用高性能网络互连，比如那些来自InfiniBand或Myrinet的网络互连。基本的网络拓扑和组织可以使用一个简单的总线拓扑，在性能很高的环境中，网状网络系统在主机之间提供较短的潜伏期，所以可改善总体网络性能和传输速率。
- 一般不区分高性能计算、超级计算、并行计算之间的差别。

(a) 广州超算：天河-2

(b) 天津超算：天河-1A

(c) 深圳超算：曙光星云

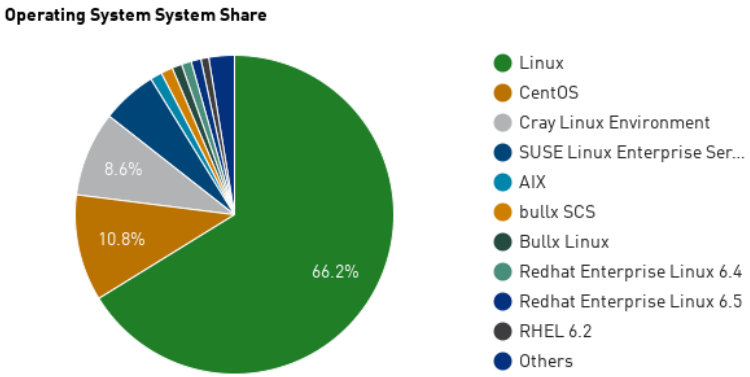
(d) 济南超算：神威蓝光

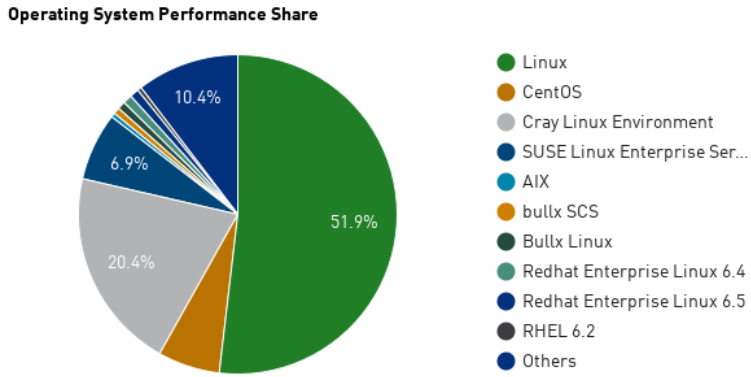
影响高性能计算的主要因素

- 硬件：
  - CPU：
    - 主要参数：主频、核数、并发数、Cache
    - 评测程序：SPEC、HPL(High Performance Linpack)
  - 内存：
    - 主要参数：大小、主频、CL延迟
    - 评测程序：Stream
  - IO能力：
    - 主要参数：缓存、转速、接口速率
    - 评测程序：IOZone、dd
  - 网络：
    - 主要参数：带宽、延迟
    - 评测程序：IMB(Intel MPI Benchmark)
  - 能耗：
- 软件：
  - 编译器、数值函数库、并行库、作业调度、监控
- 设置：
  - 硬件
  - 操作系统
  - 软件

2015年11月的TOP500中Linux份额

- 高性能计算系统排名：http://www.top500.org
- 除了6套UNIX（AIX）外，其余全为Linux





OPERATING SYSTEM	COUNT	SYSTEM SHARE (%)	RMAX (GFLOPS)	RPEAK (GFLOPS)	CORES
Linux	331	66.2	217,966,506	338,236,991	17,840,409
CentOS	54	10.8	26,463,884	53,381,210	2,264,324
Cray Linux Environment	43	8.6	85,874,089	119,374,016	3,408,948
SUSE Linux Enterprise Server 11	28	5.6	28,882,220	40,107,607	1,158,736
AIX	6	1.2	1,868,442	2,215,095	76,128
bullx SCS	6	1.2	2,709,920	3,497,976	129,168
Bullx Linux	5	1	3,358,042	4,299,887	125,672
Redhat Enterprise Linux 6.4	5	1	3,901,827	5,327,977	139,322
Redhat Enterprise Linux 6.5	5	1	3,839,814	5,037,773	138,004
RHEL 6.2	4	0.8	1,738,900	2,132,582	102,528
Scientific Linux	3	0.6	1,714,761	2,031,552	73,384
bullx SUperCOmputer Suite A.E.2.1	3	0.6	2,942,070	3,583,180	165,888
Redhat Enterprise Linux 6	2	0.4	2,433,470	3,032,783	295,656
Kylin Linux	2	0.4	35,934,090	57,976,934	3,294,720
Redhat Enterprise Linux 7	1	0.2	217,887	272,794	7,104
RHEL 6.1	1	0.2	230,600	340,915	37,056
SLES10 + SGI ProPack 5	1	0.2	237,800	267,878	23,040

Unix/Linux操作系统发行版本

一些主要发行版：

- Linux：
  - 常见：Android、Arch、CentOS、Debian、Fedora、Gentoo、Mandriva、Red Hat Enterprise Linux(RHEL)、Slackware、SUSE Linux Enterprise Desktop(SLED)、SUSE Linux Enterprise Server(SLES)、OpenSuSE、Ubuntu·····
  - 高性能计算系统常见：
    - RHEL系：Red Hat Enterprise Linux(RHEL)、CentOS、Scientific Linux-SL
    - SUSE系：SUSE Linux Enterprise Server(SLES)、OpenSuSE
- Unix：
  - 学院派BSD：FreeBSD、OpenBSD、NetBSD·····
  - 商业Unix：IBM AIX、HP UX、Sun Solaris、OpenSolaris[1]、Mac OS X[2]、iOS、SGI IRIX·····

Linux、BSD发布版：http://distrowatch.com/

[1] Sun公司按照CDDL授权开源

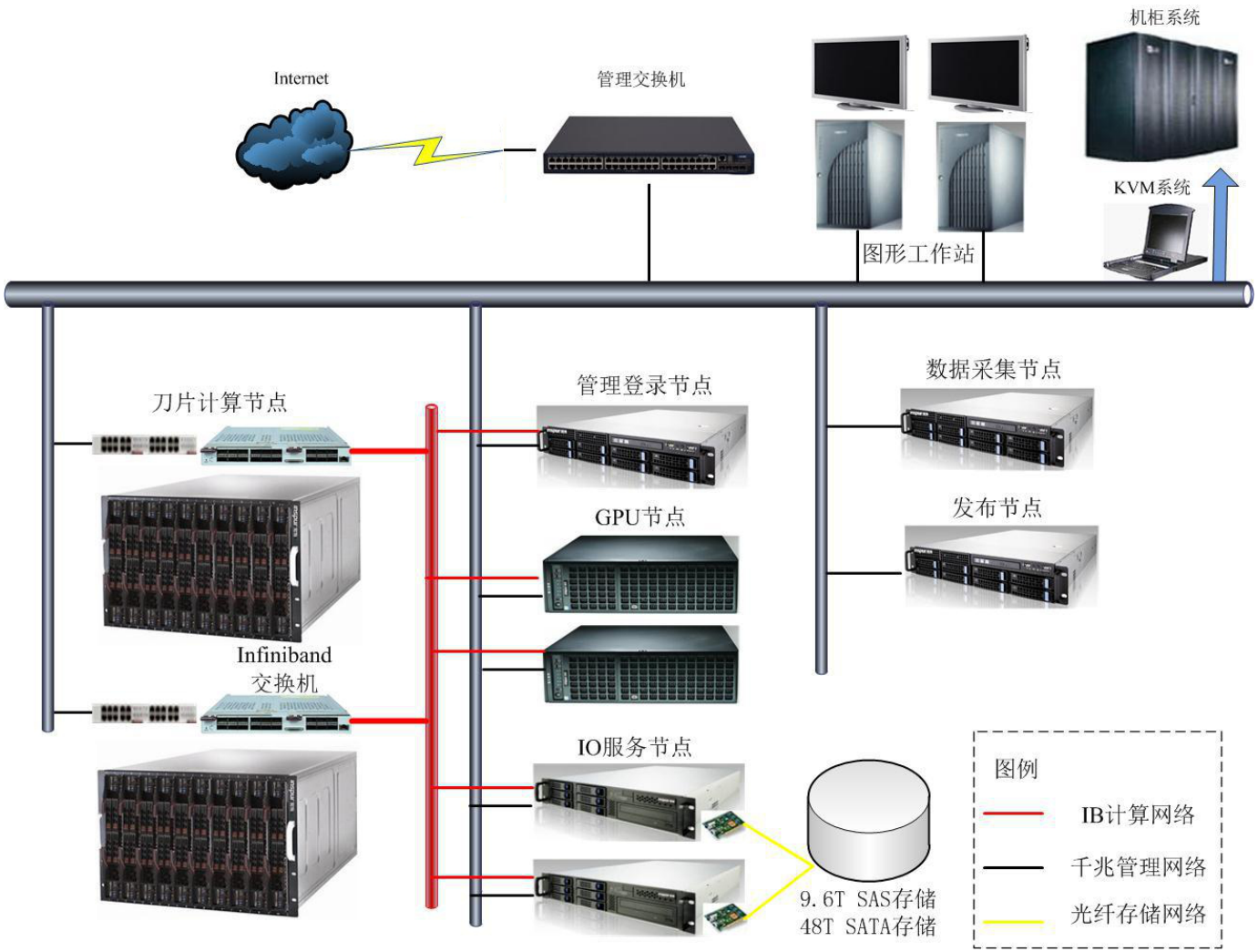
[2] 以FreeBSD源代码和Mach微内核为基础

CentOS系统

- CentOS Linux基于Red Hat Enterprise Linux(RHEL)
- 本文除非特别说明，否则都以CentOS 6 x86_64 Linux系统为例
- CentOS yum源目录：*/etc/yum.repos.d*
- CentOS常用命令：
  - 安装软件包packagename：*yum␣install␣packagename*
  - 删除软件包packagename：*yum␣erase␣packagename*
  - 搜索软件描述中含string的软件包：*yum␣search␣string*
  - 搜索哪个软件包提供文件名中含string的文件：*yum␣provides␣*string**
  - 查看软件包packagename简要信息：*yum␣list␣packagename*
  - 查看软件包packagename详细信息：*yum␣info␣packagename*
  - 列出已安装软件包：*yum␣list␣installed*
  - 查看servicename服务状态：*chkconfig␣--list␣servicename*
  - 增加servicename服务：*chkconfig␣--add␣servicename*
  - 删除servicename服务：*chkconfig␣--del␣servicename*
  - 设置系统启动时服务servicename启动：*chkconfig␣servicename␣on*
  - 设置系统启动时服务servicename关闭：*chkconfig␣servicename␣off*

注：CentOS 7.x中*service*和*chkconfig*虽还可用，但建议转到*systemctl*命令，运行*service*等时会提示代替的*systemctl*命令

集群拓扑


Internet
管理交换机
机柜系统
KVM系统
图形工作站
刀片计算节点
管理登录节点
数据采集节点
GPU节点
发布节点
Infiniband
交换机
IO服务节点
9.6T SAS存储
48T SATA存储
图例
IB计算网络
千兆管理网络
光纤存储网络

集群规划

- 主节点（管理、IO、用户登录，服务端）：
  - 节点名： admin
  - 内网eth0网卡IP： 192.168.1.254
  - 编辑*/etc/hosts*，设定节点名与IP对应：

```
#IP地址␣␣␣␣␣␣带域名的节点名␣␣␣␣␣␣␣␣␣节点名
127.0.0.1␣␣␣␣␣localhost.localdomain␣␣localhost
192.168.1.254␣admin.mydomain.org␣␣␣␣␣admin
192.168.1.1␣␣␣node1.mydomain.org␣␣␣␣␣node1
192.168.1.2␣␣␣node2.mydomain.org␣␣␣␣␣node2
```

- 计算节点（客户端）：
  - 节点名： node1、node2、...
  - 内网eth0网卡IP： 192.168.1.1、192.168.1.2、...
  - 编辑*/etc/hosts*，添加如下内容，以便能根据节点名找到主节点IP：

```
192.168.1.254␣admin.mydomain.org␣␣␣␣␣admin
```

设置SELinux及防火墙

- 关闭SELinux:
- 修改*/etc/selinux/config*，设置SELINUX=disabled，并重启系统
- 清空防火墙[3]:
  - *iptables␣-F*
  - *iptables␣-X*
  - *iptables␣-t␣nat␣-F*

[3]为了安全，iptables还是需要设置好，只允许某些IP访问某些端口等

用户初始文件

- */etc/motd*：设置登录提示
- */etc/profile.d*：放在此目录下的后缀为*.sh*和*.csh*的文件将分别被用户登录时所使用的bash或csh启动
- */etc/skel*：在此目录下的文件，开设用户账户时会自动将其放到用户主目录下，利用此可以设置初始文件方便用户使用，如：
  - *.bashrc*：bash初始文件
  - *.bash_profile*：bash初始文件
  - *.bash_logout*：用户登录退出时执行的命令，如在里面增加清除屏幕的命令：

```
if␣[␣"$SHLVL"␣=␣1␣];␣then
␣␣␣␣␣␣␣␣[␣-x␣/usr/bin/clear_console␣]␣&&␣/usr/bin/clear_console␣-q
␣␣␣␣␣␣␣␣clear
fi
```

  - *.vimrc*：设置vim配置
  - *intelmpi.pbs*：提供针对Intel MPI在PBS系统的作业脚本模板

NFS：网络文件系统

- NFS(Network File System)：网络文件系统
- 各节点需共享文件，比如*/home*和*/opt*等

NFS服务端设置

- 安装所需的NFS包：*yum -y install nfs-utils*
- 设置NFS服务系统启动时自启动：*chkconfig nfs on*
- 重启NFS服务：*service nfs restart*
- 编辑*/etc/exports*，添加NFS共享目录及允许IP

```
/home 192.168.1.0/24(rw,async,no_root_squash)
/opt  192.168.1.0/24(rw,async,no_root_squash)
```

  - 允许IP段192.168.1.0/24，别用*，否则任何地址都能连到此机，并可删除文件
  - 同步sync保证正确但降低性能，一般用异步async
- 刷新NFS设置，使其生效：*exportfs -ra*
- 查看NFS状态：*exportfs -v*

```
/home  192.168.1.0/24(rw,async,wdelay,no_root_squash,no_subtree_check)
/opt   192.168.1.0/24(rw,async,wdelay,no_root_squash,no_subtree_check)
```

- 在其他机子上运行showmount -e 该机IP地址会显示：

```
Export list for ‘IP地址‘
```

NFS客户端设置

- 编辑*/etc/fstab*，添加NFS共享目录[4]：

```
admin:/home␣/home␣nfs␣defaults,nfsvers=3␣0␣0
admin:/opt␣/opt␣nfs␣defaults,nfsvers=3␣0␣0
```

- 挂载文件系统：*mount␣-a*
- 查看挂载情况：*mount*

```
/dev/sda2 on / type ext4 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
tmpfs on /dev/shm type tmpfs (rw)
/dev/sda1 on /boot type ext3 (rw)
/dev/sda5 on /tmp type ext4 (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
admin:/home on /home type nfs (rw,nfsvers=3,addr=192.168.1.254)
admin:/opt on /opt type nfs (rw,nfsvers=3,addr=192.168.1.254)
```

[4]NFS V4版本默认采用tcp协议，导致性能降低，建议采用V3版本

NIS：网络信息服务

NIS(Network Information Service)：网络信息服务[5]

- 对主机帐号等系统信息提供集中管理的网络服务
- 用户登录任何一台NIS客户机都会从NIS服务器进行登录认证，可实现用户帐号的集中管理
- 主要同步信息：用户信息、节点名信息等
- 在NIS环境中，有三种类型的主机：
  - 服务器(master)：充当主机配置信息的中央数据库，保存着用户帐号、组帐号等配置信息的权威副本
  - 从服务器(slave)：保存这些信息的冗余副本
  - 客户机(client)：使用这些信息

[5]另一种流行的方式是采用LDAP(Lightweight Directory Access Protocol)，比NIS更强大

NIS服务端设置 I

- 安装所需包：*yum␣-y␣install␣ypserv␣ypbind␣yp-tools␣rpcbind*
- 设定NIS域名：
  - 在*/etc/sysconfig/network*中添加域名信息：

```
NISDOMAIN=mydomain.org
```

  - 设置系统启动时自动设置域名，在*/etc/rc.local*中添加：[6]

```
nisdomainname␣mydomain.org
```

  - 使当前域名生效：*nisdomainname␣mydomain.org*
  - 修改*/var/yp/securenets*，设置允许客户端IP范围：

```
host␣127.0.0.1
255.255.255.0␣192.168.1.0
```

NIS服务端设置 II

- 修改*/etc/yp.conf*[7]：

```
ypserver␣127.0.0.1
```

- 修改需要同步信息的配置文件*/var/yp/Makefile*，只同步用户信息、组信息和节点名信息：

```
all:␣␣passwd␣group␣hosts␣#rpc␣service␣netid␣protocols␣mail
```

- 初始化：*/usr/lib64/yp/ypinit␣-m*
- 启动服务：
  - rpc守护进程：*service␣rpcbind␣start*
  - 服务守护进程：*service␣ypserv␣start*
  - 客户守护进程：*service␣ypbind␣start*
  - 用户运行*yppasswd*修改密码守护进程：*service␣yppasswdd␣start*
- 设置系统启动时自启动：
  - rpc守护进程：*chkconfig␣rpcbind␣on*
  - 服务守护进程：*chkconfig␣ypserv␣on*

NIS服务端设置 III

- 客户守护进程：*chkconfig␣ypbind␣on*
- 用户运行*yppasswd*修改密码守护进程：*chkconfig␣yppasswdd␣on*

- 设置定时推送用户等信息：
  - 运行*crontab␣-e*，输入内容：

```
#␣m␣␣h␣dom␣mon␣dow␣␣␣command
␣*/5␣*␣␣*␣␣␣*␣␣␣*␣␣␣␣␣cd␣/var/yp;␣make␣>/dev/null
```

[6]防止network中设置的未生效，可以不设置
[7]服务端也可是自己的客户端

NIS客户端设置 I

- 安装所需包：*yum -y install ypbind yp-tools rpcbind*
- 保证客户端可通过服务端名字获得对应IP，修改*/etc/hosts*：

```
192.168.1.254 admin.mydomain.org admin
```

- 设定NIS域名：
  - 修改*/etc/sysconfig/network*：

```
NISDOMAIN=mydomain.org
```

  - 设置系统启动时自动设置域名，在*/etc/rc.local*中添加：

```
nisdomainname mydomain.org
```

NIS客户端设置 II

- 设定服务节点，修改*/etc/yp.conf*:

```
domain␣mydomain.org␣server␣admin
```

- 使当前域名生效：*nisdomainname␣admin*
- 设置所需要同步的信息，修改*/etc/nsswitch.conf*

```
passwd:␣files␣nis
shadow:␣files␣nis
group:␣␣files␣nis
hosts:␣␣files␣nis␣dns
```

- 启动服务：*service␣ypbind␣start*
- 启动rpc守护进程：*service␣rpcbind␣start*
- 设置系统启动时自启动rpc守护进程：*chkconfig␣rpcbind␣on*

NIS客户端设置 III

- 设置系统启动时自启动： *chkconfig␣ypbind␣on*
- 测试：
  - 检查是否启动： *ypwhich*
  - 测试用户hmli信息： *id␣hmli*
  - 检查同步的文件： *yptest*

quota：磁盘配额

- 磁盘配额就是管理员可以为用户所能使用的磁盘空间进行配额限制，每一用户只能使用最大配额范围内的磁盘空间。
- 设置磁盘配额后，可以对每一个用户的磁盘使用情况进行跟踪和控制，通过监测可以标识出超过配额报警阈值和配额限制的用户，从而采取相应的措施。
- 磁盘配额管理功能的提供，使得管理员可以方便合理地为用户分配存储资源，可以限制指定账户能够使用的磁盘空间，这样可以避免因某个用户的过度使用磁盘空间造成其他用户无法正常工作甚至影响系统运行避免由于磁盘空间使用的失控可能造成的系统崩溃，提高了系统的安全性。

磁盘配额quota设置

- 安装所需包：*yum -y install quota*
- 修改*/etc/fstab*[8]，增加选项 usrjquota=aquota.user,grpjquota=aquota.group,jqfmt=vfsv0：

```
/dev/sda5 /home ext3 defaults,usrjquota=aquota.user,grpjquota=aquota.group,jqfmt=vfsv0 1 2
```

- 重新挂载：*mount -o remount /home*
- 检查并生成所需信息[9]：*quotacheck -cvug /home*
- 开启配额：*quotaon -vug /home*
- 设置hmli用户配额：*edquota hmli*

```
Disk quotas for user hmli (uid 502):
  Filesystem                   blocks       soft       hard     inodes     soft     hard
  /dev/sda5                  13907432   49000000   50000000      23491        0
0
```

格式说明：

文件系统 当前占用块大小 块大小软限制 块大小硬限制 当前inodes大小 inodes软限制 inodes硬限制

- 查看hmli用户磁盘配额：*quota hmli*
- 关闭磁盘配额：*quotaoff -vug /home*

[8]假设/dev/sda5为需要设置的分区

[9]启动quota前必须做

网络批量系统安装

大规模的部署Red Hat Linux系（CentOS等）操作系统

- 避免手工安装的繁琐
- 避免出错，保证一致性

网络批量系统安装

大规模的部署Red Hat Linux系（CentOS等）操作系统

- 避免手工安装的繁琐
- 避免出错，保证一致性
- dd硬盘或nc网络对拷对拷：
  - 1->2->4->8->...
  - 需要拔插硬盘或光盘或网络启动、挂载分区、修改IP地址、节点名等，繁琐

网络批量系统安装

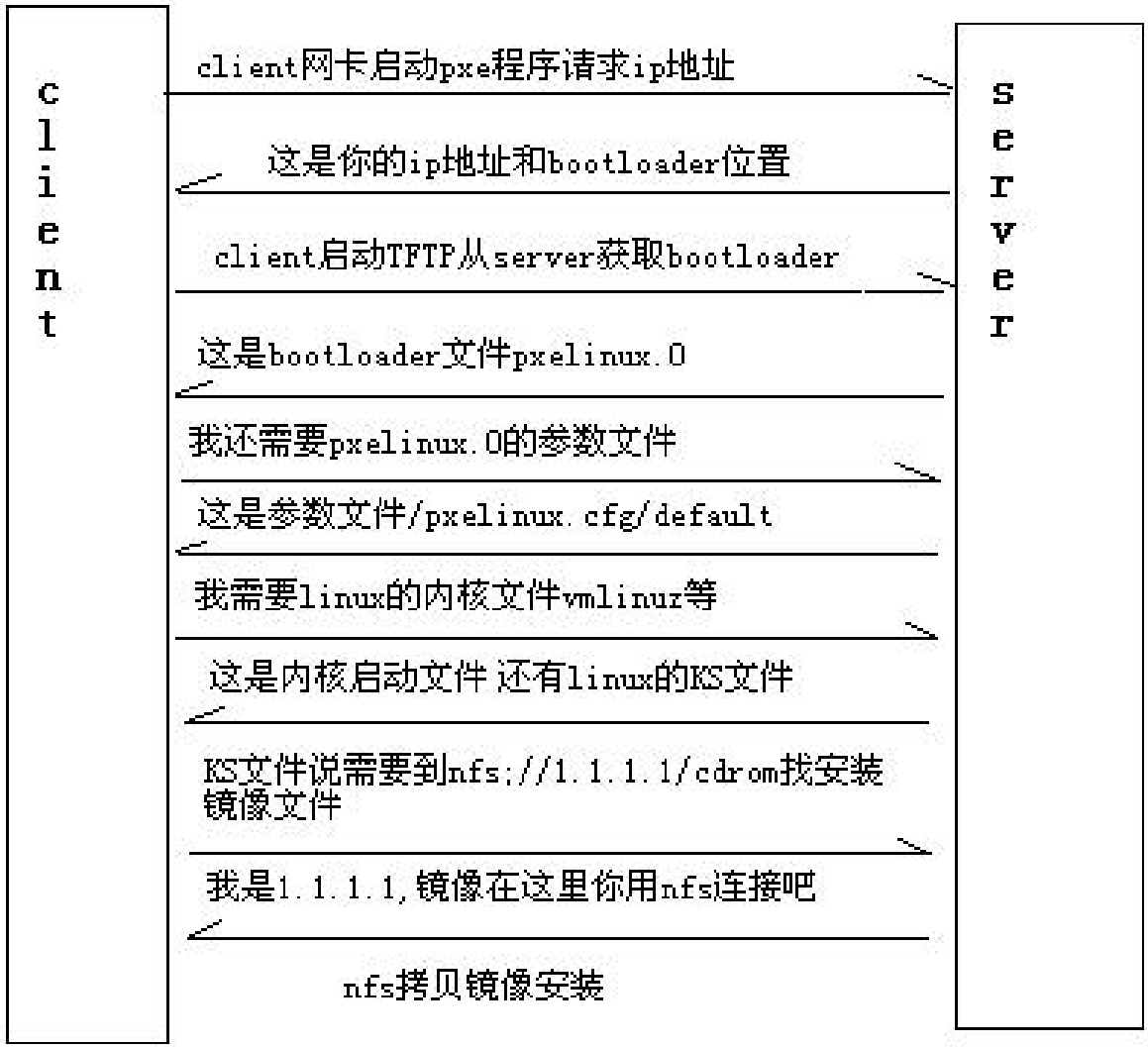
大规模的部署Red Hat Linux系（CentOS等）操作系统

- 避免手工安装的繁琐
- 避免出错，保证一致性
- dd硬盘或nc网络对拷对拷：
  - 1->2->4->8->...
  - 需要拔插硬盘或光盘或网络启动、挂载分区、修改IP地址、节点名等，繁琐
- kickstart网络安装：
  - 许多系统管理员宁愿使用自动化的安装方法来安装Red Hat Linux系
  - 为满足这种需要，Red Hat创建了kickstart安装
  - 使用kickstart，系统管理员可创建一个文件，这个文件包含了在典型的安装过程中所遇到问题的答案
  - kickstart文件可以存放于单一的服务器上，在安装过程中被独立的机器所读取
  - 此安装方法可以支持使用单一kickstart文件在多台机器上安装Red Hat Linux，对于网络和系统管理员来说是个理想的选择
  - kickstart给用户提供了一种自动化安装装Red Hat Linux的方法
  - CentOS等Red Hat系Linux发行版也支持kickstart安装

PXE网络安装 I

- PXE(Pre-boot Execution Environment)是由Intel设计的协议，它可以使计算机通过网络启动
- 协议分为client和server两端，PXE client在网卡的ROM中，当计算机引导时，BIOS把PXE client调入内存执行，并显示出命令菜单，经用户选择后，PXE client将放置在远端的操作系统通过网络下载到本地运行
- 在其启动过程中，客户端请求服务器分配IP地址，之后PXE Client使用TFTP Client通过TFTP(Trivial File Transfer Protocol)协议下载启动安装程序所需的文件
- PXE网络安装：客户机通过支持PXE的网卡向网络中发送请求DHCP信息的广播请求IP地址等信息，DHCP服务器给客户端提供IP地址和其它信息（TFTP服务器、启动文件等），之后请求并下载安装需要的文件

PXE网络安装 II


client
server
client网卡启动pxe程序请求ip地址
这是你的ip地址和bootloader位置
client启动TFTP从server获取bootloader
这是bootloader文件pxelinux.0
我还需要pxelinux.0的参数文件
这是参数文件/pxelinux.cfg/default
我需要linux的内核文件vmlinuz等
这是内核启动文件 还有linux的KS文件
KS文件说需要到nfs://1.1.1.1/cdrom找安装镜像文件
我是1.1.1.1,镜像在这里你用nfs连接吧
nfs拷贝镜像安装

DHCP安装和配置 I

- 安装dhcp包： *yum -y install dhcp*
- 复制模板文件：
  cp /usr/share/doc/dhcp-4.1.1/dhcpd.conf.sample /etc/dhcp/dhcpd.conf
- 修改*/etc/dhcp/dhcpd.conf*，设置PXE文件及IP与MAC地址的对应：

```
# option definitions common to all supported networks...
option domain-name "mydomain.org";#域名
option domain-name-servers ns1.ustc.edu.cn;#域名服务器

default-lease-time 600;
max-lease-time 7200;

subnet 192.168.1.0 netmask 255.255.255.0 {
        option routers              192.168.1.254;#路由网关
        option subnet-mask          255.255.255.0;
        option nis-domain           "mydomain.org";#域名
        option domain-name          "mydomain.org";
        option domain-name-servers 192.168.1.254;#域名服务器

        option time-offset          -18000;#Eastern Standard Time
        range dynamic-bootp 192.168.1.1 192.168.1.243;#bootp IP范围
        default-lease-time 21600;
```

DHCP安装和配置 II

```
        max-lease-time 43200;
        host node1 {
                hardware ethernet 40:61:86:ED:95:30; #IP地址与MAC地址对应
                fixed-address 192.168.1.1;
        }
        host node2 {
                hardware ethernet 40:61:86:ed:93:7e;
                fixed-address 192.168.1.2;
        }
}
option space PXE;
class "pxeclients" {
        match if substring(option vendor-class-identifier, 0, 9) = "PXEClient";
        next-server 192.168.1.254; #PXE服务器
        filename "/pxelinux.0"; #PXE文件
}
```

- 如不知道MAC地址，那么将会自动分配随机地址，客户端系统装好后可以修改客户端配置设置成固定IP
- 启动DHCP服务：*service dhcp start*
- 设置系统启动时自启动服务：*chkconfig dhcp on*

配置TFTP服务器

PXE安装时，客户机使用TFTP协议从服务器下载引导文件并执行

- 安装配置TFTP服务器：*yum -y install tftp-server*
- tftp服务由xinetd服务管理，编辑 */etc/xinetd.d/tftp*：

```
# default: off
# description: The tftp server serves files using the trivial file transfer \
#       protocol.  The tftp protocol is often used to boot diskless \
#       workstations, download configuration files to network-aware printers, \
#       and to start the installation process for some operating systems.
service tftp
{
        socket_type     = dgram
        protocol        = udp
        wait            = yes
        user            = root
        server          = /usr/sbin/in.tftpd
        server_args     = -s /tftpboot #目录名
        disable         = no #为： no
        per_source      = 11
        cps             = 100 2
        flags           = IPv4
}
```

- 重启服务：*service xinetd restart*

PXE引导配置 I

- PXE启动映像文件由syslinux软件包提供，CentOS镜像中已提供，以下以ISO镜像中为例
- 安装syslinux以获取*pxelinux.0*等：*yum␣-y␣install␣syslinux*
- 将*pxelinux.0*复制到*/tftpboot*：*cp␣/usr/share/syslinux/pxelinux.0␣/tftpboot*
- 挂载第一张DVD镜像：
  mount␣-o␣loop␣CentOS-6.7-x86_64-bin-DVD1.iso␣/mnt
- 将安装光盘目录中的启动文件复制*/tftpboot*
  cp␣/mnt/images/pxeboot/{vmlinuz,initrd.img}␣/tftpboot
- 创建存放客户端的配置文件*default*：
  mkdir␣/tftpboot/pxelinux.cfg;␣cp␣/mnt/isolinux/isolinux.cfg␣/tftpboot/pxelinux.cfg/de

PXE引导配置 II

- 修改配置*/tftpboot/pxelinux.cfg/default*:

```
default␣linux
#prompt␣1#␣不要提示，直接进行安装
timeout␣60␣#提示时的等待时间

display␣boot.msg

menu␣background␣splash.jpg
menu␣title␣Welcome␣to␣CentOS␣6.7!

label␣linux
␣␣␣␣␣␣␣␣menu␣label␣^Install␣or␣upgrade␣an␣existing␣system
␣␣␣␣␣␣␣␣menu␣default
␣␣␣␣␣␣␣␣kernel␣vmlinuz
␣␣␣␣␣␣␣␣append␣initrd=initrd.img
␣␣␣␣␣␣␣␣append␣ksdevice=eth0␣load_ramdisk=1␣initrd=initrd.img␣network␣\
␣␣␣␣␣␣␣␣␣␣␣␣ks=nfs:192.168.1.254:/tftpboot/ks.cfg␣noipv6␣devfs=nomount␣selinux=0␣nostorage␣\
␣␣␣␣␣␣␣␣␣␣␣␣driverload=sd_mod:mptbase:mptscsih:mptsas:aacraid:mptscsih:libata:megaraid_sas:scsi_mod
#主要为上面三行，设置ks文件，内核引导参数等
label␣local
␣␣␣␣␣␣␣␣menu␣label␣Boot␣from␣^local␣drive
␣␣␣␣␣␣␣␣localboot␣0
```

配置NFS目录

将*/mnt*通过NFS共享给客户端：

- 编辑*/etc/exports*，增加下面一行：

```
/mnt 192.168.1.0/24(rw,async,no_root_squash)
```

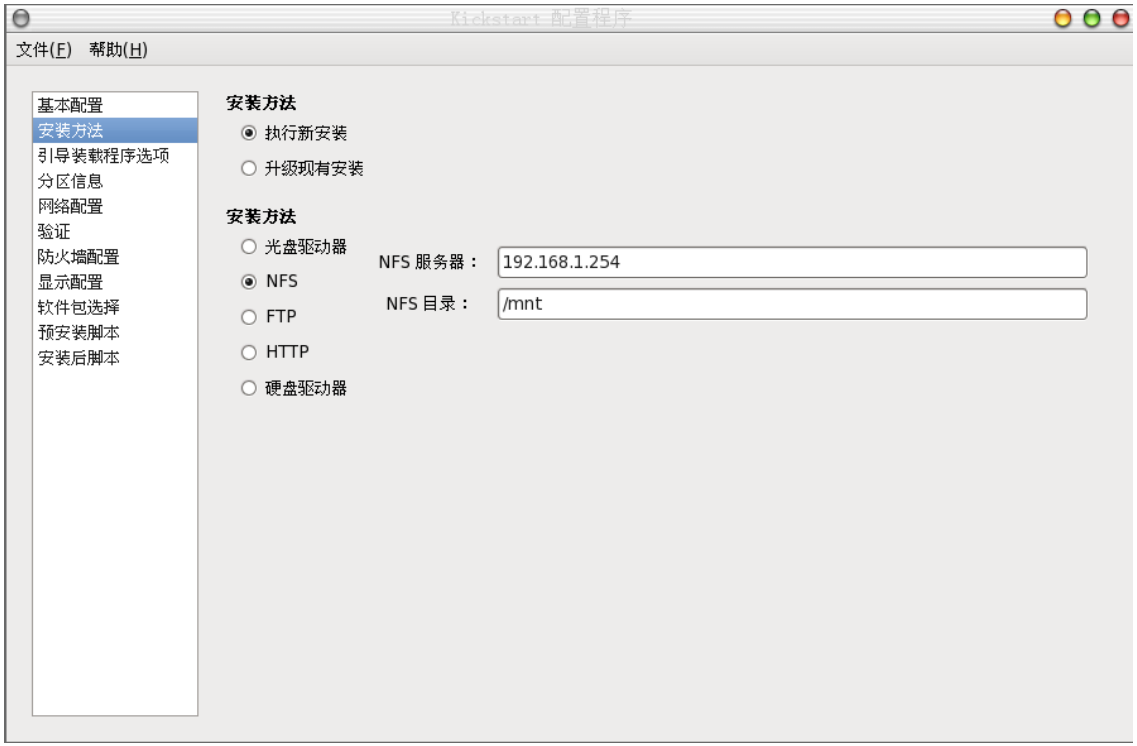
- 刷新NFS配置：*exportfs -ra*

配置Kickstart安装 I

- 通常，在安装操作系统的过程需大量的人机交互过程，为减少交互过程，为提高安装效率，Red Hat Linux开始支持称为kickstart的功能，
- 只需事先定义好一个kickstart自动应答配置文件（通常存放在安装服务器上），并让安装程序知道该配置文件的位置，在安装过程中安装程序就可以自己从该文件中读取安装配置，这样就避免了繁琐的人机交互，实现无人值守的自动化安装
- 安装好一台CentOS机器，安装程序都会创建一个kickstart配置文件 */root/anaconda-ks.cfg*，记录真实安装配置

配置Kickstart安装 II

- 使用*system-config-kickstart*命令配置*ks.cfg*文件：
  - 安装所需包：*yum␣-y␣install␣system-config-kickstart*
  - 运行图形界面进行设置：*system-config-kickstart*
  - 载入*/root/anaconda-ks.cfg*作为模板：文件->打开文件，选择*/root/anaconda-ks.cfg*
  - 在此界面上进行配置并保存为*/tftpboot/ks.cfg*



配置Kickstart安装 III

- 修改*/tftpboot/ks.cfg*，增减所需安装的软件包等：

```
#platform=x86,␣AMD64或Intel␣EM64T
#version=DEVEL
#␣Firewall␣configuration
firewall␣--disabled
#␣Install␣OS␣instead␣of␣upgrade
install
#␣Use␣NFS␣installation␣media
nfs␣--server=192.168.1.254␣--dir=/mnt␣#设置所需要下载文件的协议及服务端IP和目录
#␣Root␣password
rootpw␣--iscrypted␣$6$fQW/BV8Uw/af07Vh$YnA8l5jZtVSvAOEKMkzpriC6pl/ntXnrRK1cPZgsZSS0qv6v6sGjdxR.qc8zC15C
#␣System␣authorization␣information
auth␣␣--useshadow␣␣--passalgo=sha512␣--enablenis␣--nisdomain=mydomain.org␣--nisserver=admin
#␣Use␣text␣mode␣install
text
#␣System␣keyboard
keyboard␣us
#␣System␣language
lang␣en_US
#␣SELinux␣configuration
selinux␣--disabled
#␣Do␣not␣configure␣the␣X␣Window␣System
skipx
#␣Installation␣logging␣level
```

配置Kickstart安装 IV

```
logging␣--level=info

#␣System␣timezone
timezone␣␣Asia/Shanghai
#␣Network␣information
network␣␣--bootproto=dhcp␣--device=eth0␣--onboot=on
#␣System␣bootloader␣configuration
bootloader␣--location=mbr
#␣Clear␣the␣Master␣Boot␣Record
zerombr
#␣Partition␣clearing␣information
clearpart␣--all␣--initlabel
#␣Disk␣partitioning␣information␣#␣设置硬盘分区
part␣/boot␣--fstype=”ext3”␣--size=200
part␣/␣--fstype=”ext4”␣--size=10000
part␣swap␣--fstype=”swap”␣--size=6000
part␣/tmp␣--fstype=”ext4”␣--grow␣--size=1000

%packages␣#␣增减所需要安装的软件包
@base
@network-server
@performance
@storage-server
@system-admin-tools
cpufrequtils
```

```
dhcp
sdparm
yp-tools
tree
tuned
tuned-utils
vim-enhanced
ypbind
-lvm2
-nano
-pcmciautils
-plymouth
-rfkill
-rsync
-system-config-firewall-tui
-system-config-network-tui
-unzip
-vconfig
-wireless-tools
-vim-minimal
%end
```

客户端网络启动安装

- 设置客户端BIOS，选择从网卡启动，具体方法因BIOS版本不同而异
- 网卡中的PXE代码会联系DHCP服务器来获取IP地址以及启动镜像，然后启动镜像被载入并运行
- 安装完成后，安装程序会提示你重新启动机器
- 重新启动机器时切记要在BIOS里改成从硬盘启动
- 如仍然从光盘启动机器，又会重复前面的自动安装步骤

ssh免输密码访问

- MPI并行程序运行时，需要无需输入密码，因此需要配置ssh或rsh免输密码访问[10]
- ssh可采用：
  - 基于密钥：适合用户自己设置
  - 基于主机：适合系统管理员设置，无需用户自己设置

[10]建议ssh，rsh一般不再使用

ssh免输密码访问：基于密钥

用户主目录*/home*是共享时，用户使用自己的账户运行以下命令：

- 生成密钥：*ssh-keygen*
- 追加到*~/.ssh/authorized_keys*：
  cat␣~/.ssh/id_rsa.pub␣>>~/.ssh/authorized_keys
- 设置权限，禁止他人可读写：
  - *chmod␣700␣~/.ssh*
  - *chmod␣600␣~/.ssh/id_rsa*
- 以下与集群设置无关，只是告知一个方法方便访问linux系统：
  - 可以设置在node1节点上远程免输密码访问node2节点：
  - 在node1上：*ssh-copy-id␣username@node2*

注意：私钥id_rsa一定要保护好，否则别人拿到了，就可用它访问上述在authorized_keys中加了公钥id_rsa.pub的系统

ssh免输密码访问：基于主机 I

采用root账户设置，设置后其他非root用户无需自己设置即可免输密码访问（root用户不支持）

- *ssh-keyscan*获取各节点rsa，生成*/etc/ssh/ssh_known_hosts*：

```
#!/bin/sh
NODES='admin'
for i in `seq 1 100` #生成100个节点的，注意：`为键盘左上角的反引号`，不是单引号'
do
        NODES=$NODES" "node$i
done
ssh-keyscan -t rsa $NODES | sort | sed -e 's/^node\([[:digit:]]*\)/node\1,192.168.1.\1/' >>/etc/ssh/ssh_known_hosts
sed -i 's/^admin/admin,192.168.1.254/' /etc/ssh/ssh_known_hosts
```

生成的*ssh_known_hosts*文件类似：

```
admin,192.168.1.254 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQ9+IMIpGEQAyviMpZtpfN7PrFtwM1Rox/b+...==
node1,192.168.1.1 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAEQAyviMQ9+IMIpGpZtpfN7Pr1RoxFtwM/b+...==
node2,192.168.1.2 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAEQAwrn8pOQZjucFkQjhg8iYhT1rDtU8jANmZ+...==
```

ssh免输密码访问：基于主机 II

- 生成*/etc/ssh/shosts.equiv*:

```
#!/bin/sh
echo admin >>/etc/ssh/shosts.equiv
for i in `seq 1 100`
do
        echo node$i >>/etc/ssh/shosts.equiv
done
```

生成的*shosts.equiv*文件类似:

```
admin
node1
node2
```

ssh免输密码访问：基于主机 III

- 编辑*/etc/ssh/sshd_config*:

```
HostbasedAuthentication␣yes
IgnoreRhosts␣no
```

- 编辑*/etc/ssh/ssh_config*:

```
Host␣*
␣␣␣␣HostbasedAuthentication␣yes
␣␣␣␣EnableSSHKeysign␣yes
```

- 可在主节点上设置，设置好后复制这四个文件到其它节点*/etc/ssh*下
- 各节点重启ssh服务：*service␣sshd␣restart*

NTP：时间服务

- Network Time Protocol(NTP)：网络时间协议
- 目的：同步各节点时间，保持各节点的时间一致性
- ntpdate与ntpd的不同：
  - ntpdate：立即将客户端与授时服务器的时间同步，不会考虑其他程序是否会阵痛，直接调整时间
  - ntpd：在实际同步时间时是一点点的校准过来时间的，最终把时间慢慢的校正对

NTP服务端设置

服务端设置:

- 安装NTP服务: *yum -y install ntp ntpdate*
- 对时: *ntpdate time.ustc.edu.cn*
- 编辑*/etc/ntp.conf*:

```
# Hosts on local network are less restricted.
restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap #设置客户端范围
# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
server time.ustc.edu.cn #科大时间服务器
server 0.rhel.pool.ntp.org
server 1.rhel.pool.ntp.org
server 2.rhel.pool.ntp.org
```

- 设置系统启动时启动: *chkconfig ntpd on*
- 重启服务: *service ntpd restart*
- 查看服务状态: *ntpq -p*

NTP客户端设置

- 安装NTP包：*yum -y install ntp ntpdate*
- 对时：*ntpdate admin*
- 编辑*/etc/ntp.conf*:

```
restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap #设置客户端范围
server admin #主节点服务器
server 0.rhel.pool.ntp.org
server 1.rhel.pool.ntp.org
server 2.rhel.pool.ntp.org
```

- 设置系统启动时启动：*chkconfig ntpd on*
- 服务端重启ntpd服务后，在一定时间内客户端运行*ntpdate admin*时出现下述信息，请等待一会儿再测试:

```
18 Nov 14:59:29 ntpdate[10863]: no server suitable for synchronization found.
```

- 查看服务状态：*ntpq -p admin*

```
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*netfee.ustc.edu 195.113.144.201  2 u   16   64   77    0.233   -4.352   1.593
+dns2.synet.edu  118.143.17.82    2 u    9   64   77   38.886   -2.280   0.947
```

内网客户端访问外网：iptables转发

客户端通过服务端连接外网进行升级及软件安装等

- 服务端，假设eth1为外网网卡，eth0为内网网卡，运行下述脚本：

```
#!/bin
echo␣1␣>␣/proc/sys/net/ipv4/ip_forward␣#打开转发

modprobe␣ip_conntrack_ftp␣#本机FTP时需用
modprobe␣ip_nat_ftp␣␣#通过本机FTP需用
iptables␣-t␣nat␣-A␣POSTROUTING␣-j␣SNAT␣-o␣eth1␣--to␣eth1-IP␣#注意为eth1及eth1␣IP
```

- 客户端只要设置为默认走与服务端的内网IP即可，类似：
  ip␣route␣add␣default␣via␣192.168.1.254

VSFTP简介

- vsftpd是“very secure FTP daemon”的缩写，安全性是它的一个最大的特点
- 支持很多其他的 FTP 服务器所不支持的特征。比如：非常高的安全性需求、带宽限制、良好的可伸缩性、可创建虚拟用户、支持IPv6、速率高等
- 是一款在Linux发行版中最受推崇的FTP服务器程序。特点是小巧轻快，安全易用

FTP两种模式

- FTP服务器两种通道
  - 命令通道：21端口
  - 数据传输通道
- 主动模式（PORT）（Server->Client）
  - 客户端打开端口N（N为>1024的随机端口）连接服务器21端口建立命令通道
  - 客户端通过N+1端口与服务器20端口建立数据传输通道
- 被动模式（PASV）（Client->Server）
  - 客户端打开端口N（N为>1024的随机端口）连接服务器21端口建立命令通道（同上）
  - 客户端通过N+1端口与服务器>1024随机端口建立数据传输通道，客户端使用PASV命令
- 推荐方式：使用被动模式传输可以尽最大可能降低因客户端防火墙配置导致的超时问题

VSFTPD配置

- 安装：*yum -y install vsftpd*
- 修改配置文件*/etc/vsftpd/vsftpd.conf*:

```
anonymous_enable=NO #不允许匿名登录
local_enable=YES  #允许本地用户
#chroot_local_user=YES #本地用户是否chroot，即只能在自己目录夏
#chroot_list_enable=YES #是否允许chroot用户列表
#chroot_list_file=/etc/vsftpd/chroot_list #允许chroot的用户列表文件
#以下有些选项为为设置pasv模式，可以不设置，使用主动模式
pam_service_name=vsftpd #pam验证名
userlist_enable=YES
tcp_wrappers=YES
use_localtime=YES
pasv_promiscuous=YES #pasv模式
pasv_address=211.86.151.104 #pasv模式地址，本机地址
pasv_enable=YES #启用pasv模式
pasv_min_port=10000 #pasv模式最小端口号
pasv_max_port=10100 #pasv模式最大端口号
max_per_ip=10 #每个IP对多10个连接
```

- 设置自启动：*chkconfig vsftpd on*
- 重启服务：*service vsftpd restart*

ulimit

- ulimit的可以限制用户使用的CPU、内存、stack等
  - *ulimit -a*可以查看全部设置
  - 内存或进程数等不做限制的话，有可能会导致系统响应缓慢甚至死机
  - stack太小的话，有可能会导致某些程序运行出错
- 设置*/etc/security/limits.conf*:

```
* soft memlock 66060288 #64GB内存
* hard memlock 66060288
* soft stack unlimited #stack不做限制
* hard stack unlimited
```

注:

- 系统设置的hard限制，用户无法超过
- 系统设置的soft限制，用户可以超过

设置密码过期时间：login.defs

- 修改配置文件*/etc/login.defs*:

```
PASS_MAX_DAYS␣150␣#密码的最长有效时限，150天
PASS_MIN_DAYS␣0␣#允许密码的最短有效时限
PASS_MIN_LEN␣␣5␣#允许的最小密码长度，5位
PASS_WARN_AGE␣7␣#密码过期前多少天开始提醒用户，提前7天提醒
```

- 此文件修改后仅对之后生成的账户起作用

设置密码有效期：chage

*chage*命令可为用户设置密码期限等

- 语法：*chage␣[options]␣[LOGIN]*
- 主要选项：
  - -d, –lastday LAST_DAY 设置密码最后修改时间
  - -E, –expiredate EXPIRE_DATE 设置密码过期时间
  - -h, –help 显示帮助
  - -I, –inactive INACTIVE 设置密码过期后账户停止
  - -l, –list 显示账号信息
  - -m, –mindays MIN_DAYS 设置密码的最短有效时限
  - -M, –maxdays MAX_DAYS 设置密码的最长有效时限
  - -W, –warndays WARN_DAYS 设置在密码过期前多少天开始提醒用户

设置密码过期及锁定账户等：usermod与passwd

- *usermod*命令
  - 语法：*passwd␣[options]␣[LOGIN]*
  - 选项：
    - -e, –expiredate EXPIRE_DATE 设置账户过期时间
    - -f, –inactive INACTIVE 设置密码过期后账户停止
    - -L, –lock 禁用账户
    - -U, –unlock 解禁账户
- *passwd*命令：
  - 语法：*usermod␣[options]␣LOGIN*
  - 选项：
    - -k, –keep-tokens 保持身份验证令牌不过期
    - -d, –delete 删除已命名帐号的密码
    - -l, –lock 禁用账号
    - -u, –unlock 解禁帐号
    - -e, –expire 让账号过期
    - -x, –maximum=DAYS 密码的最长有效时限
    - -n, –minimum=DAYS 密码的最短有效时限
    - -w, –warning=DAYS 在密码过期前多少天开始提醒用户
    - -i, –inactive=DAYS 当密码过期后经过多少天该帐号会被禁用

复杂密码设置

- 安装相应包：*yum␣-y␣install␣pam_passwdqc.x86_64*
- 修改*/etc/pam.d/system-auth*：

```
#password␣requisite␣pam_cracklib.so␣try_first_pass␣retry=3␣type=␣#原有，用下面行代替
password␣requisite␣pam_passwdqc.so␣min=disabled,12,8,6,5␣max=40␣passphrase=3␣match=4␣similar=deny␣random=42␣\
enforce=everyone␣retry=3
```

pam_passwdqc提供了非常多选项，上述这些规则其本质是：

- 不接受任何单种字符类的口令
- 对两种字符混合的密码，强制口令最小长度是12位
- 对口令字强制最小长度是8位
- 对3种字符混合的密码强制最小长度是6位
- 4种字符混合的密码强制最小长度是5位
- 所谓4种字符混合的密码就是由“数字”，“小写字母”，“大写字母”，以及“其它字符”组成（其它字符就是类似“!”、“_”这种）
- 强制任何口令长度不得超过40位

iptables防火墙设置

```
#!/bin/sh
echo␣1␣>␣/proc/sys/net/ipv4/ip_forward␣#允许转发，允许内网访问外网
modprobe␣ip_conntrack_ftp␣#加载模块，允许内网访问外网
modprobe␣ip_nat_ft␣#加载模块，允许内网访问外网
iptables␣-F␣#清空规则，默认为filter表
iptables␣-t␣nat␣-F␣#清除nat表中的规则链
iptables␣-t␣nat␣-X␣#删除nat表
iptables␣-t␣nat␣-A␣POSTROUTING␣-j␣SNAT␣-o␣eth1␣--to␣211.86.151.100␣#设置转发，允许内网访问外网
iptables␣-X␣USTC␣#删除USTC规则链
iptables␣-N␣USTC␣#新建USTC规则链
iptables␣-A␣USTC␣-j␣ACCEPT␣-s␣202.38.64.0/19␣#允许此类地址访问主机
iptables␣-A␣USTC␣-j␣ACCEPT␣-s␣210.45.64.0/20
iptables␣-A␣USTC␣-j␣ACCEPT␣-s␣210.45.112.0/20
iptables␣-A␣USTC␣-j␣ACCEPT␣-s␣211.86.144.0/20
iptables␣-A␣USTC␣-j␣ACCEPT␣-s␣222.195.64.0/19
iptables␣-A␣USTC␣-j␣ACCEPT␣-s␣202.141.160.0/20
iptables␣-A␣USTC␣-j␣ACCEPT␣-s␣114.214.160.0/19
iptables␣-A␣USTC␣-j␣ACCEPT␣-s␣114.214.192.0/18
iptables␣-X␣BLK␣#清除BLK规则链
iptables␣-N␣BLK␣#新建BLK规则链
iptables␣-A␣BLK␣-j␣DROP␣-s␣210.45.72.1␣#禁止此IP访问
iptables␣-A␣BLK␣-j␣DROP␣-s␣210.45.77.0/24
iptables␣-A␣INPUT␣-j␣USTC␣-i␣eth1␣-p␣tcp␣--dport␣21:22␣#允许USTC规则链中的访问21:22端口
iptables␣-A␣INPUT␣-m␣state␣--state␣RELATED,ESTABLISHED␣-j␣ACCEPT
iptables␣-I␣INPUT␣-p␣tcp␣--dport␣10000:10100␣-j␣ACCEPT␣#与vsftpd.conf中设置的端口范围一致
iptables␣-A␣INPUT␣-i␣eth1␣␣-m␣limit␣--limit␣1/sec␣-j␣LOG␣--log-prefix␣"Firewall:␣"␣#记录日志，并添加"Firewall:␣"前缀
iptables␣-A␣INPUT␣-j␣DROP␣-i␣eth1␣#禁止其它访问eth1
```

fail2ban设置

- fail2ban:
  - 主要防止从外部采用暴力尝试密码攻击，比如某个IP短时间内连续5次输入SSH密码错误，自动封锁此IP 10分钟等
  - 只需在对外的节点上设置
- 设置:
  - fail2ban是在EPEL源中，首先需要设置好源:
    - 下载配置放到*/etc/yum.repos.d*中:

      wget␣https://lug.ustc.edu.cn/wiki/_export/code/mirrors/help/epel?codeblock=0␣-O␣/etc/yum.repos.d/epel.repo
    - 下载GPG-KEY放到*/etc/pki/rpm-gpg*中:

      wget␣http://mirrors.ustc.edu.cn/epel/RPM-GPG-KEY-EPEL-6␣-O␣/etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
  - 安装: *yum␣-y␣install␣fail2ban*
  - 修改*/etc/fail2ban/jail.conf*和*/etc/fail2ban/fail2ban.conf*等打开需要监控的服务
  - 重启fail2ban服务: *service␣fail2ban␣restart*
  - 设置服务自启动: *chkconfig␣fail2ban␣on*

rkhunter

rkhuner检查可疑文件等

- 安装：*yum␣-y␣install␣rkhunter*
- 检查：*rkhunter␣-c*
- 更新数据库：*rkhunter␣--update*

检查包文件是否被修改

- 命令：
  - *rpm␣-V␣包名*
  - *rpm␣-Vf␣文件名*
- 无输出：表示没修改
- 有输出：表示对应文件被修改过

```
S.5....T.    c /etc/profile
S.5....T.    c /etc/securetty
```

- 输出含义：

S	文件大小不一致	c %config	配置文件
M	权限不一致	d %doc	文档文件
5	md5值不一致	g %ghost	幽灵文件，如包中没包含的文件
D	设备major/minor数不一致	l %license	license文件
L	readLink(2)路径不一致	r %readme	readme文件
U	用户不一致		
G	用户组不一致		
T	时间不一致		
P	包含文件不同		

psacct

- 安装：*yum -y install psacct*
- 设置自启动：*chkconfig psacct on*
- 重启：*service psacct restart*
- 主要命令：

ac	输出用户登录/退出（连接时间，数小时）的统计信息
lastcomm	输出用户之前执行的命令的信息
accton	用于开启/关闭进程会计机制（process accounting）
sa	概述之前执行的命令的信息

集群批量处理

多台机器如何管理?

集群批量处理

多台机器如何管理?

- 一台台登录上去：繁琐、容易出错、速度慢

集群批量处理

多台机器如何管理?

- 一台台登录上去：繁琐、容易出错、速度慢
- 利用for循环、Expect脚本等处理：需要自己编写

集群批量处理

多台机器如何管理?

- 一台台登录上去：繁琐、容易出错、速度慢
- 利用for循环、Expect脚本等处理：需要自己编写
- 集群管理软件：省时省力

集群批量处理

多台机器如何管理?

- 一台台登录上去：繁琐、容易出错、速度慢
- 利用for循环、Expect脚本等处理：需要自己编写
- 集群管理软件：省时省力
- 常见集群管理软件：pdsh、mussh、synctool、xcat、clusterssh、The Cluster Command and Control(C3)

pdsh：并行SHELL

- pdsh：http://sourceforge.net/projects/pdsh/
- CentOS官方源不含pdsh，需添加第三方epel源，参见：http://lug.ustc.edu.cn/wiki/mirrors/help/epel
- 所有节点都安装：*yum␣-y␣install␣pdsh*
- 用法：
  - 并行执行命令：*pdsh␣-w␣node[1-100]␣-x␣node[81-90]␣hostname*
  - 并行复制到远端：*pdcp␣-w␣node1,node[12-18]␣intel.sh␣/etc/profile.d/*
  - 并行从远端复制到本地，本地名自动加.节点名区分：*rpdcp␣-w␣node1,node12␣/etc/profile.d/intel.sh␣/tmp*

Expect：交互式脚本语言

- 管理员利器，交互式脚本语言：Expect
- 基于Tcl语言

```
#!/usr/bin/expect
for { set i 1 } { $i < 101 } { incr i } {
        set node node$i
        spawn scp -r .ssh $node:
        expect "connecting"; send "yes\r"
        expect "password"; send "yourpassword\r"
        expect "root"; puts "over"
}
```

编译环境

- C/C++、FORTRAN编译器（支持OpenMP）:
  - GCC
  - Intel
  - PGI
  - NAG
- MPI环境:
  - Intel MPI
  - Open MPI
  - MPICH、MPICH2
  - MVAPICH、MVAPICH2
  - HP MPI
  - LAM MPI
  - IBM MPI

C/C++/FORTRAN编译环境

- GCC：*yum␣-y␣install␣gcc␣gcc-c++␣gcc-gfortran*
- Intel：
  - 下载编译器压缩包，解压缩后到解压缩后类似目录*parallel_studio_xe_2016*下执行*./install.sh*
  - 当前登录设置环境变量：*../opt/intel/parallel_studio_xe_2016.0.047/bin/psxevars.sh␣intel64*
  - 设置默认登录后的环境变量，编辑*/etc/profile.d/intel.sh*：[11]

```
../opt/intel/parallel_studio_xe_2016.0.047/bin/psxevars.sh␣intel64
```

[11] 放置在*/etc/profile.d*下后缀为.sh或.csh的文件自动被bash或csh执行

Open MPI编译环境

- Open MPI安装：
  - 下载：http://www.open-mpi.org/software/ompi/v1.8/downloads/openmpi-1.8.1.tar.bz2
  - 解压缩：*tar␣xvf␣openmpi-1.8.1.tar.bz2*
  - 配置：

    FC=ifort␣CC=icc␣CXX=icpc␣./configure␣--prefix=/opt/openmpi-1.8.1
  - 编译：*make*
  - 安装：*make␣install*
- 配置环境变量，编辑*/etc/profile.d/openmpi.sh*：

```
OPENMPI=/opt/openmpi-1.8.1
if␣test␣-z␣"`echo␣$PATH␣|␣grep␣$OPENMPI/bin`";␣then␣#␣PATH
␣␣␣␣␣␣␣␣PATH=$OPENMPI/bin:${PATH}
␣␣␣␣␣␣␣␣export␣PATH
fi
if␣test␣-z␣"`echo␣$LD_LIBRARY_PATH␣|␣grep␣$OPENMPI/lib`";␣then␣#␣LD_LIBRARY_PATH
␣␣␣␣␣␣␣␣LD_LIBRARY_PATH=$OPENMPI/lib${LD_LIBRARY_PATH:+:}${LD_LIBRARY_PATH}
␣␣␣␣␣␣␣␣export␣LD_LIBRARY_PATH
fi
if␣test␣-z␣"`echo␣$MANPATH␣|␣grep␣$OPENMPI/share/man`";␣then␣#␣MANPATH
␣␣␣␣␣␣␣␣MANPATH=$OPENMPI/share/man:${MANPATH}
␣␣␣␣␣␣␣␣export␣MANPATH
fi
```

mpi-selector管理MPI环境 I

- 搜寻https://www.rpmfind.net下载mpi-selector包装上
- 在MPI安装后的*bin*目录下设置配置文件，设置*PATH*和*LD_LIBRARY_PATH*等环境变量
  - 针对BASH的*mpivars.sh*:

```
# PATH
source /opt/intel/composer_xe_2015.1.133/bin/compilervars.sh intel64
OPENMPI=/opt/openmpi/1.8.2_intel-compiler-2015.1.133
if test -z "`echo $PATH | grep $OPENMPI/bin`"; then
        PATH=$OPENMPI/bin:${PATH}
        export PATH
fi
# LD_LIBRARY_PATH
if test -z "`echo $LD_LIBRARY_PATH | grep $OPENMPI/lib`"; then
        LD_LIBRARY_PATH=$OPENMPI/lib${LD_LIBRARY_PATH:+:}${LD_LIBRARY_PATH}
        export LD_LIBRARY_PATH
fi
# MANPATH
if test -z "`echo $MANPATH | grep $OPENMPI/share/man`"; then
        MANPATH=$OPENMPI/share/man:${MANPATH}
        export MANPATH
fi
```

mpi-selector管理MPI环境 II

- 针对CSH的*mpivars.csh*:

```
source␣/opt/intel/composer_xe_2015.1.133/bin/compilervars.csh␣intel64
set␣OPENMPI=/opt/openmpi/1.8.2_intel-compiler-2015.1.133
#␣path
if␣(""␣==␣"‘echo␣$path␣|␣grep␣$OPENMPI/bin‘")␣then
␣␣␣␣␣␣␣␣set␣path=($OPENMPI/bin␣$path)
endif
#␣LD_LIBRARY_PATH
if␣("1"␣==␣"$?LD_LIBRARY_PATH")␣then
␣␣␣␣␣␣␣␣if␣("$LD_LIBRARY_PATH"␣!~␣*$OPENMPI/lib*)␣then
␣␣␣␣␣␣␣␣␣␣␣␣setenv␣LD_LIBRARY_PATH␣$OPENMPI/lib:${LD_LIBRARY_PATH}
␣␣␣␣␣␣␣␣endif
else
␣␣␣␣␣␣␣␣setenv␣LD_LIBRARY_PATH␣$OPENMPI/lib
endif
#␣MANPATH
if␣("1"␣==␣"$?MANPATH")␣then
␣␣␣␣␣␣␣␣if␣("$MANPATH"␣!~␣*$OPENMPI/share/man*)␣then
␣␣␣␣␣␣␣␣␣␣␣␣setenv␣MANPATH␣$OPENMPI/share/man:${MANPATH}
␣␣␣␣endif
else
␣␣␣␣␣␣␣␣setenv␣MANPATH␣$OPENMPI/share/man:
endif
```

mpi-selector管理MPI环境 III

- 注册：

mpi-selector␣--register␣openmpi-1.8.2_intel-compiler-15.1.133␣--source-dir␣/opt/openmpi/1.8.2_intel-compiler-15.1.133/bin

- 选择默认环境：*mpi-selector-menu*选择一个合适的，并设置为系统默认：

```
Current system default: intel-mpi-5.0.2.044_intel-compiler-15.1.133
Current user default:
    "u" and "s" modifiers can be added to numeric and "U"
    commands to specify "user" or "system-wide".
1. intel-mpi-5.0.1.035_intel-compiler-15.0.090
2. intel-mpi-5.0.2.044_intel-compiler-15.1.133
3. intel-mpi-5.1.1.109_intel-compiler-16.0.109
4. openmpi-1.6.5_intel-compiler-15.0.090
5. openmpi-1.6.5_intel-compiler-15.1.133
6. openmpi-1.6.5_pgi-14.10
7. openmpi-1.8.2_intel-compiler-15.0.090
8. openmpi-1.8.2_intel-compiler-15.1.133
9. openmpi-1.8.2_pgi-14.10
U. Unset default
Q. Quit
Selection (1-11[us], U[us], Q): 2s
```

mpi-selector管理MPI环境 IV

- 将配置复制到计算节点：
  - 复制默认环境：*scp␣/etc/sysconfig/mpi-selector␣node1:/etc/sysconfig*
  - 复制MPI环境配置仓库：
    scp␣-r␣/var/mpi-selector/data␣node1:/var/mpi-selector

使用modules管理编译运行环境 I

- 安装：*yum -y install environment-modules*
- 创建配置：
  - 建立目录：*mkdir /etc/modules/intel*
  - 建立配置文件*2015.1.133*，设置*PATH*等环境变量：

```
#%Module1.0
proc ModulesHelp { } {
global dotversion
puts stderr ``\tIntel Compiler 2015.1.133 (icc, icpc, ifort)''
}
conflict intel/2015.0.090 #冲突配置文件
module-whatis ``Intel Compiler 2015.1.133 (icc, icpc, ifort)''
prepend-path PATH /opt/intel/composer_xe_2015.1.133/bin/intel64
prepend-path LD_LIBRARY_PATH /opt/intel/composer_xe_2015.1.133/compiler/lib/intel64
prepend-path LIBRARY_PATH /opt/intel/composer_xe_2015.1.133/compiler/lib/intel64
prepend-path MANPATH /opt/intel/composer_xe_2015.1.133/man
setenv CC icc
setenv CXX icpc
setenv FC ifort
setenv F77 ifort
setenv F90 ifort
```

注：以Tcl格式书写，由modulecmd命令处理

使用modules管理编译运行环境 II

- 其它节点同步：*scp␣-r␣/etc/modules/intel␣node1:/etc/modules*
- 常用命令：
  - 查看可用的module：*module␣avail*
  - 查看当前使用的module：*module␣list*
  - 加载module：
    *module␣load␣MODULE_NAME*或*module␣add␣MODULE_NAME*
  - 卸载module：
    *module␣unload␣MODULE_NAME*或*module␣rm␣MODULE_NAME*
  - 显示module内容：*module␣disp␣MODULE_NAME*
  - 切换module：*module␣switch␣OLD_MODULE␣NEW_MODULE*
  - 卸载所有已加载module：*module␣purge*
  - 显示module说明：*module␣whatis␣[MODULE_NAME]*

作业调度系统

- 在一个大型系统内部，通常需要处理一些自动化运行的任务，通常会采用系统自带的crontable的定时任务完成
- 但是，很多情况下，是多个作业，彼此先后执行，共同完成任务。在这样的情况下，定时任务存在两个明显的问题：
  - 浪费了大量的系统等待时间
  - 假设两个作业，第一个作业必须在第二个作业前运行，如果第二个作业先运行，就会有灾难性的后果，对于定时任务而言，解决任务这样两个作业优先级的问题是只能把任务一的运行时间安排在二之前，不能完全满足前面的假设，但是对于作业调度器而言，安排作业的优先级，是最基本的功能，简直是小Case
- 常见作业调度系统：Condor、LSF、PBS(PBS Pro、OpenPBS、TORQUE、曙光GridView、浪潮TSJM、联想LJRS)、Maui、Moab、SGE(Sun Grid Engine、Oracle Grid Engine)、SLURM

TORQUE和Maui

- 主页：http://www.adaptivecomputing.com/products/open-source/
- TORQUE资源管理器：
  - 6.x：增加支持cgroup、禁止nodes被*qmgr*动态编辑、支持使用提高的权限运行作业开始者脚本
  - 5.x：增加支持CPU频率控制、qrerun all命令、节点能耗控制
  - 4.2.x：增加对Intel Xeon Phi的支持
  - 4.1.x：针对Cray系统，不建议使用
  - 4.0.x：增强千万亿次系统的支持
  - 3.0.x：在2.5系列基础上增加了对NUMA架构的支持
  - 2.5.x：
- Maui集群调度：可与多种资源管理器配合，调度策略优，如TORQUE，负责作业调度

TORQUE服务端安装

- 解压缩：*tar␣xvf␣torque-6.1.2.tar.gz*
- 进入目录：*cd␣torque-6.1.2*
- 配置：*./configure␣--prefix=/opt/torque/6.1.2*
  注：
  - 如需支持GPU或MIC，需要打开对应选项：*--enable-nvidia-gpus*或*--enable-mic*
  - 如需支持cpuset或cgroups，需要打开*--enable-cpuset*或*--enable-cgroups*，cgroups（含cpuset）打开后会忽略cpuset选项，另外需要额外安装libhwloc>=1.9.1的包，CentOS 7自带的版本太低
- 编译：*make*
- 安装：*make␣install*
- 编辑*/etc/profile.d/torque.sh*：

```
TORQUE=/opt/torque/6.1.2
if␣[␣"`id␣-u`"␣-eq␣0␣];␣then
␣␣␣␣PATH=$PATH:$TORQUE/bin:$TORQUE/sbin
else
␣␣␣␣PATH=$PATH:$TORQUE/bin
fi
MANPATH=$MANPATH:$TORQUE/man
export␣PATH␣MANPATH
```

TORQUE服务端配置 I

- 在解压缩后的源文件目录*/opt/src/torque-6.1.2*下
- 设置trqauthd服务：
  - CentOS 6.x：
    - 复制服务脚本：*cp␣contrib/init.d/trqauthd␣/etc/init.d/*
    - 增加到系统服务：*chkconfig␣--add␣trqauthd*
    - 设置开机自启动：*chkconfig␣trqauthd␣on*
    - 启动服务：*service␣trqauthd␣start*
  - CentOS 7.x：
    - 复制服务脚本：*cp␣contrib/systemd/trqauthd.service␣/lib/systemd/system/*

      新版本安装时已自动设置好，如没，则需要执行此步
    - 增加到系统自启动服务：*systemctl␣enable␣trqauthd*
    - 启动服务：*systemctl␣restart␣trqauthd*
- 初始化数据库：*./torque.setup␣root*

  新版本安装时已自动设置好，如没，则需要执行此步

TORQUE服务端配置 II

- 添加计算节点的机器名与对应的核数(np)，编辑 */var/spool/torque/server_priv/nodes*：

```
#node-name[:ts] [np=] [gpus=] [properties]
#node5 np=2 cluster01 rackNumber26 RAM16GB
node1 np=12 gpus=2
node2 np=8
```

注：也可用*qmgr -c 'set server auto_node_np = True'*设置自动设置CPU核数，将覆盖掉上面配置

- 设置pbs_server服务
  - CentOS 6.x：
    - 复制服务脚本：*cp contrib/init.d/pbs_server /etc/init.d*

      新版本安装时已自动设置好，如没，则需要执行此步
    - 增加到系统服务：*chkconfig --add pbs_server*
    - 设置开机自启动：*chkconfig pbs_server on*
    - 启动服务：*service pbs_server restart*

TORQUE服务端配置 III

- CentOS 7.x：
  - 复制服务脚本：*cp␣contrib/systemd/pbs_server.service␣/lib/systemd/system/*

    新版本安装时已自动设置好，如没，则需要执行此步
  - 增加到系统自启动服务：*systemctl␣enable␣pbs_server*
  - 启动服务：*systemctl␣restart␣pbs_server*
- 创建队列：
  - 生成server：*pbs_server␣-t␣create*

    新版本安装时已自动设置好，如没，则需要执行此步
  - 设置队列：*qmgr*，输入下面Qmgr:后的内容，将设置一个默认队列batch：

```
Qmgr: create queue batch queue_type=execution
Qmgr: set server default_queue=batch
Qmgr: set queue batch started=true
Qmgr: set queue batch enabled=true
Qmgr: set server scheduling=true
```

- 日志目录：*/var/spool/torque/server_logs*

TORQUE测试

- 终止：*qterm -t quick*
- 启动：
  - CentOS 6.x：*service pbs_server start*
  - CentOS 7.x：*systemctl start pbs_server*
- 查看队列：*qstat -q*
- 查看配置：*qmgr -c 'p s'*
- 查看节点信息：*pbsnodes -a*
- 提交作业测试：*echo "sleep 30" | qsub*

计算节点安装TORQUE

- 服务端:
  在解压缩后的目录*/opt/src/torque-6.1.2*下运行*make␣packages*
  生成:
  - *torque-package-clients-linux-x86_64.sh*
  - *torque-package-devel-linux-x86_64.sh*
  - *torque-package-doc-linux-x86_64.sh*
  - *torque-package-mom-linux-x86_64.sh*
  - *torque-package-server-linux-x86_64.sh*
- 计算节点:
  将*torque-package-clients-linux-x86_64.sh*、*torque-package-mom-linux-x86_64.sh*复制到计算节点，然后执行:
  - *./torque-package-clients-linux-x86_64.sh␣--install*
  - *./torque-package-mom-linux-x86_64.sh␣--install*

计算节点配置TORQUE I

- 设置服务端主节点名（可选，默认已设置好）：
  echo␣"admin"␣>/var/spool/torque/server_name
- 编辑*/var/spool/torque/mom_priv/config*

```
$pbsserver␣␣␣␣␣␣admin␣␣␣␣␣#服务端主机名
$logevent␣␣␣␣␣␣␣255␣␣␣␣␣␣␣#日志级别
$usecp␣admin:/home␣/home␣␣#对NFS共享目录采用cp而不是scp复制文件
$spool_as_final_name␣true␣#作业正常和出错时屏幕输出
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣#即时存储到其工作目录日志文件，
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣#否则会在作业结束时才存储在文件中
```

- 复制主节点的*/etc/profile.d/torque.sh*到计算节点的*/etc/profile.d*下：
  scp␣admin:/etc/profile.d/torque.sh␣/etc/profile.d

计算节点配置TORQUE II

- 在解压缩后的源文件目录*/opt/src/torque-6.1.2*下
- 设置pbs_mom服务
  - CentOS 6.x:
    - 复制服务脚本：*cp␣contrib/init.d/pbs_mom␣/etc/init.d*

      新版本安装时已自动设置好，如没，则需要执行此步
    - 增加到系统服务：*chkconfig␣--add␣pbs_mom*
    - 设置开机自启动：*chkconfig␣mom␣on*
    - 启动服务：*service␣pbs_mom␣restart*
  - CentOS 7.x:
    - 复制服务脚本：*cp␣contrib/systemd/pbs_mom.service␣/lib/systemd/system/*

      新版本安装时已自动设置好，如没，则需要执行此步
    - 增加到系统子启动服务：*systemctl␣enable␣pbs_mom*
    - 启动服务：*systemctl␣restart␣pbs_mom*
  - 日志目录：*/var/spool/torque/mom_logs*

计算节点配置TORQUE III

注意：InfiniBand、OPA等对资源需求大，建议在*pbs_mom*中添加下面设置，否则容易出现无法打开这些设备的错误：

- CentOS 6.x：
  文件*/etc/init.d/pbs_mom*：

```
ulimit -s unlimited
ulimit -m unlimited
ulimit -l unlimited
```

- CentOS 7.x：
  文件*/lib/systemd/system/pbs_mom.service*：

```
LimitNOFILE=32768
LimitMEMLOCK=infinity
LimitSTACK=infinity
```

Maui安装

- Maui比TORQUE自带的集群调度器更好
- Maui只在服务端安装，计算节点无需安装
  - 解压缩：*tar␣xvf␣maui-3.3.1.tar.gz*
  - 进入目录：*cd␣maui-3.3.1*
  - 配置：*./configure␣--prefix=/opt/maui/3.3.1␣--with-pbs=/opt/torque/6.1.2*
  - 编译：*make*
  - 安装：*make␣install*

服务节点上配置Maui

- 编辑*/usr/local/maui/maui.cfg*:

```
SERVERHOST␣␣␣␣␣␣␣␣␣␣␣␣admin
#␣primary␣admin␣must␣be␣first␣in␣list
ADMIN1␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣root
#␣Resource␣Manager␣Definition
RMCFG[admin]␣TYPE=PBS@RMNMHOST@
RMTYPE[0]␣PBS
```

注：在这里可以设置队列、用户权限策略等，详见：
http://docs.adaptivecomputing.com/maui/

- 编辑*/etc/profile.d/maui.sh*设置环境变量:

```
MAUI=/opt/maui/3.3.1
if␣[␣"‘id␣-u‘"␣-eq␣0␣];␣then
␣␣␣␣␣␣␣␣PATH=$PATH:$MAUI/bin:$MAUI/sbin
else
␣␣␣␣␣␣␣␣PATH=$PATH:$MAUI/bin
fi
```

- 信息记录:
  - 日志：*/usr/local/maui/log*
  - 统计信息：*/usr/local/maui/stats*

启动Maui

- 注意不要在服务节点上启动*pbs_sched*，如已启动，则先终止
- 启动Maui：*/opt/maui/3.3.1/sbin/maui*
- 设置系统启动时自启动，在*/etc/rc.local*中添加：

```
/opt/maui/3.3.1/sbin/maui
```

对于CentOS 7.x需要执行以下操作以确保/etc/rc.local启动时自动执行：

- *chmod␣+x␣/etc/rc.d/rc.local*
- *systemctl␣enable␣rc-local*

PBS作业脚本

```
#!/bin/sh
#An␣example␣for␣Intel␣MPI␣job.
#DO␣NOT␣RUN␣THIS␣SCRIPT␣DIRECTLY,
#PLEASE␣RUN␣THIS␣SCRIPT␣WITH␣qsub:␣qsub␣intelmpi_job.pbs
#
#PBS␣-N␣job_name
#PBS␣-o␣job.log
#PBS␣-e␣job.err
#PBS␣-q␣normal
#PBS␣-l␣nodes=2:ppn=12
cd␣$PBS_O_WORKDIR
echo␣Begin␣Time:␣‘date‘
echo␣Directory␣is␣$PWD
echo␣This␣job␣run␣on␣the␣nodes:
cat␣$PBS_NODEFILE
mpirun␣./intelmpi-prog
echo␣End␣Time:␣‘date‘
```

作业日志目录：主节点*/var/spool/torque/job_logs*

记账

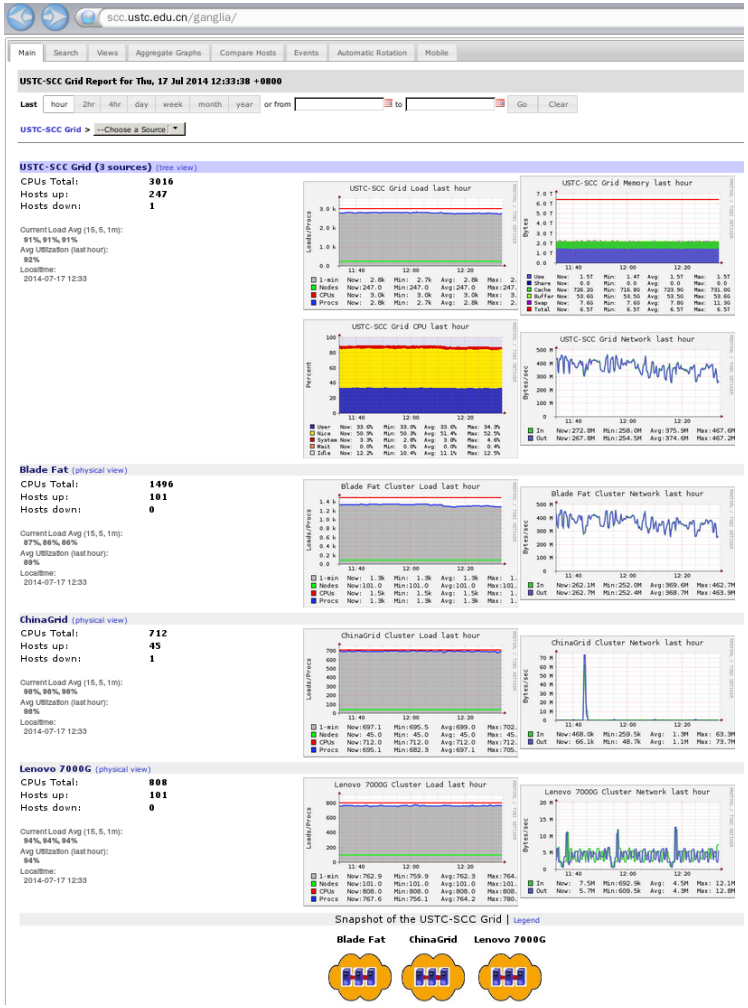
- 记账目录：主节点*/var/spool/torque/server_priv*
- 记录类型：

  A：abort 作业被服务器终止
  C：checkpoint 作业被checkpoint后保持
  D：delete 作业被删除
  E：exit 作业退出（正常或异常退出）
  Q：queue 作业被提交或入队
  R：rerun 尝试重新运行作业
  S：start 尝试开始作业
  T：restart 尝试重启作业

- 记账变量

  ctime：作业生成时间
  etime：作业获得资格开始运行时间
  qtime：作业入队时间
  start：作业开始时间
  end：作业结束时间

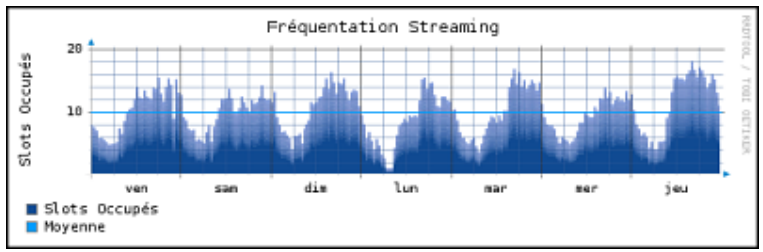
集群监控Ganglia



http://scc.ustc.edu.cn/ganglia/

集群监控Ganglia

- Ganglia：
  - 主要是用来监控系统性能的软件，如：cpu、mem、硬盘利用率、I/O负载、网络流量情况等
  - 通过曲线很容易见到每个节点的工作状态，对合理调整、分配系统资源，提高系统整体性能起到重要作用
  - Ganglia是分布式的监控系统，有两个Daemon：
    - 客户端Ganglia Monitoring Daemon(gmond)
    - 服务端Ganglia Meta Daemon(gmetad)
  - Ganglia Web：Ganglia基于PHP开发和运行的web统计浏览程序
- 依赖于：
  - PHP：基于服务端来创建动态网站的脚本语言，可生成网站主页
  - RRDtool：系统存放和显示time-series（网络带宽、温度、人数、服务器负载等），且可绘出有用的图表用来显示处理的数据和数据密度



- 官方站点：
  - http://ganglia.sourceforge.net/

Ganglia安装

CentOS官方源不含Ganglia，需添加第三方epel源，参见：
http://lug.ustc.edu.cn/wiki/mirrors/help/epel

- 服务端：*yum␣-y␣install␣ganglia-gmetad␣ganglia-gmond␣ganglia-web*
- 客户端：*yum␣-y␣install␣ganglia-gmond*

Ganglia客户端配置

- 编辑*/etc/ganglia/gmond.conf*:

```
cluster␣{
␣␣␣␣␣␣␣␣name␣=␣"mycluster1"␣#设置源，与服务端一致
␣␣␣␣␣␣␣␣owner␣=␣"HMLi"
␣␣␣␣␣␣␣␣latlong␣=␣"N31.84␣E117.27"
␣␣␣␣␣␣␣␣url␣=␣"http://scc.ustc.edu.cn"
}
host␣{
␣␣␣␣␣␣␣␣location␣=␣"5,0,2␣#节点位置，各节点不同，格式为␣Rack,␣Rank␣and␣Plane
}
udp_send_channel␣{
␣␣␣␣␣␣␣␣#bind_hostname␣=␣yes␣#Highly␣recommended,␣soon␣to␣be␣default.
␣␣␣␣␣␣␣␣mcast_if␣=␣eth0␣#发送信息的网卡，如为eth0，可省略
␣␣␣␣␣␣␣␣mcast_join␣=␣239.2.11.71
␣␣␣␣␣␣␣␣port␣=␣8649
␣␣␣␣␣␣␣␣ttl␣=␣1
}
udp_recv_channel␣{
␣␣␣␣␣␣␣␣mcast_if␣=␣eth0␣#收集信息的网卡，如为eth0，可省略
␣␣␣␣␣␣␣␣mcast_join␣=␣239.2.11.71
␣␣␣␣␣␣␣␣port␣=␣8649
␣␣␣␣␣␣␣␣bind␣=␣239.2.11.71
}
```

Ganglia服务端配置

- 编辑*/etc/ganglia/gmond.conf*，内容与客户端基本一致
- 编辑*/etc/ganglia/gmetad.conf*:

```
data_source␣"mycluster1"␣localhost␣#数据源名字，只收集客户端cluster段中name同名的
#data_source␣"mycluster2"␣ip␣#支持多个源，每行设置一个数据源的名字及IP地址
gridname␣"myGrid1"␣#网格名
all_trusted␣on␣#允许收集其它节点的
```

- 配置httpd，编辑*/etc/httpd/conf.d/ganglia.conf*:

```
Alias␣/ganglia␣/usr/share/ganglia
<Location␣/ganglia>␣#␣可按照自己需要设置允许的␣IP␣访问范围
␣␣␣␣␣␣␣␣Order␣allow,deny
␣␣␣␣␣␣␣␣allow␣from␣all
␣␣␣␣␣␣␣␣#Order␣deny,allow
␣␣␣␣␣␣␣␣#Deny␣from␣all
␣␣␣␣␣␣␣␣#Allow␣from␣127.0.0.1
␣␣␣␣␣␣␣␣#Allow␣from␣::1
␣␣␣␣␣␣␣␣#Allow␣from␣.example.com
</Location>
```

启动服务与测试

- 服务端：
  - 启动客户守护进程：*service␣gmond␣start*
  - 启动服务守护进程：*service␣gmetad␣start*
  - 启动httpd进程：*service␣httpd␣start*
  - 系统启动自启动客户守护进程：*chkconfig␣gmond␣on*
  - 系统启动自启动启动服务守护进程：*chkconfig␣gmetad␣on*
  - 系统启动自启动启动httpd守护进程：*chkconfig␣httpd␣on*
- 客户端：
  - 启动客户守护进程：*service␣gmond␣start*
  - 系统启动自启动启动服务守护进程：*chkconfig␣gmond␣on*
- 测试：*telnet␣admin␣8649␣|␣grep␣\<HOST*[12]应有类似输出：

```
<HOST NAME=”node28” IP=”192.168.1.28” REPORTED=”1353728304” TN=”1” TMAX=”20” DMAX=”0”
LOCATION=”2,8,0” GMOND_STARTED=”1353725964”>
<HOST NAME=”node46” IP=”192.168.1.46” REPORTED=”1353728305” TN=”0” TMAX=”20” DMAX=”0”
LOCATION=”4,6,0” GMOND_STARTED=”1353725965”>
Connection closed by foreign host.
```

- web访问：http://地址/ganglia/

[12]admin也可为其它节点名

出错时处理

- 查看日志：*/var/log*、……
- 根据错误信息寻找解决办法
- *man*相关命令及配置
- 搜索

联系信息

- 中国科大超算中心:
  - 电话：0551-63602248
  - 信箱：sccadmin@ustc.edu.cn
  - 主页：http://scc.ustc.edu.cn
  - 办公室：中国科大东区新图书馆一楼东侧126室
- 李会民:
  - 电话：0551-63600316
  - 信箱：hmli@ustc.edu.cn
  - 主页：http://hmli.ustc.edu.cn
  - 办公室：中国科大东区新科研楼A座二楼204室