机群系统搭建实验

调研报告

2014211304 班

D 组

史文翰 2014211218

林宇辰 2014211223

王剑督 2014211228

崔嘉伟 2014211233

杨　莹 2014211238

徐丹雅 2014211243

郝绍明 2014210123

分工情况

王剑督 2014211228 **崔嘉伟** 2014211233：

负责配置环境和测试运行

杨　莹 2014211238：

负责实验报告的编写和整理

林宇辰 2014211223 **史文翰** 2014211218 **徐丹雅** 2014211243

郝绍明 2014210123：

负责查阅资料等辅助性工作

一、 实验目的

掌握机群系统的关键组件，掌握机群系统的基本原理。

二、 实验环境

操作系统：Microsoft Windows 10 家庭中文版（x64）
虚拟机：VMware® Workstation 12 Pro
Linux： Ubuntu x64

三、 实验步骤

从机部分（node2,node3,node4，以 node4 建立过程举例）

(一) 配置前的准备工作

配置/etc/hosts 文件，该文件可以实现 IP 地址和机器的对应解析，所有节点的该文件均要按下面的内容修改：

```
wjd@ubuntu:~$ ifconfig
ens33     Link encap:Ethernet  HWaddr 00:0c:29:c2:33:64
          inet addr:10.201.8.200  Bcast:10.201.8.223  Mask:255.255.255.224
          inet6 addr: 2001:da8:215:c146:8969:1694:e60b:5540/64 Scope:Global
```

```
wjd@ubuntu:~$ cat /etc/hosts
10.201.8.195 node1
10.201.8.199 node2
10.201.8.206 node3
10.201.8.200 node4
127.0.0.1       localhost
127.0.1.1       ubuntu

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
wjd@ubuntu:~$
```

通过以上配置后节点之间能够通过各节点的机器名称相互访问。例如，可以通过 ping node2 进行测试。注意事项：

该测试必须在关闭 Linux 防火墙的条件下进行，否则可能失败。

```
wjd@ubuntu:~$ ping node3
PING node3 (10.201.8.206) 56(84) bytes of data.
64 bytes from node3 (10.201.8.206): icmp_seq=1 ttl=64 time=55.6 ms
64 bytes from node3 (10.201.8.206): icmp_seq=2 ttl=64 time=6.44 ms
64 bytes from node3 (10.201.8.206): icmp_seq=3 ttl=64 time=3.89 ms
64 bytes from node3 (10.201.8.206): icmp_seq=4 ttl=64 time=5.02 ms
64 bytes from node3 (10.201.8.206): icmp_seq=5 ttl=64 time=8.57 ms
64 bytes from node3 (10.201.8.206): icmp_seq=6 ttl=64 time=9.62 ms
64 bytes from node3 (10.201.8.206): icmp_seq=7 ttl=64 time=36.4 ms
64 bytes from node3 (10.201.8.206): icmp_seq=8 ttl=64 time=8.14 ms
64 bytes from node3 (10.201.8.206): icmp_seq=9 ttl=64 time=2.00 ms
64 bytes from node3 (10.201.8.206): icmp_seq=10 ttl=64 time=16.7 ms
64 bytes from node3 (10.201.8.206): icmp_seq=11 ttl=64 time=4.78 ms
64 bytes from node3 (10.201.8.206): icmp_seq=12 ttl=64 time=9.85 ms
64 bytes from node3 (10.201.8.206): icmp_seq=13 ttl=64 time=5.30 ms
```

(二) 挂载 NFS 文件系统

由于 MPICH 的安装目录和用户可执行程序在并行计算时需要在所有节点保存副本，而且目录要相互对应，每次一个节点一个节点的复制非常麻烦，采用 NFS 文件系统后可以实现所有节点内容与主节点内容同步更新，并自动实现目录的对应。NFS 文件系统使得所有机器都能以同样的路径访问服务器上保存的文件，访问方法如同对本地文件的访问。通常我们会将 MPICH 的安装目录及并行程序存放目录配置为 NFS 共享目录，这样可以省去将文件向各个节点复制的麻烦，大大提高工作效率。

NFS 文件系统的配置方法示例如下（假设 NFS 服务器 IP 为 192.168.1.2，配置需要在 root 用户下完成）。

1. 服务器端配置方法（下面的配置只在主节点进行）。
   1) /etc/exports 文件配置
      在文件/etc/exports 中增加以下几行：
      /usr/cluster 192.168.1.3(rw,sync,no_root_squash,no_subtree_check)
      /usr/cluster 192.168.1.4(rw,sync,no_root_squash,no_subtree_check)
      这几行文字表明 NFS 服务器向 IP 地址为 192.168.1.3,192.168.1.4 的 2 个节点共享其 /usr/cluster 目录（目录必须存在），并使这些节点具有相应的权限（可查询相关的文档）。如有更多的节点可按此方法填写。
   2) 启动 NFS 服务
      启动 NFS 服务只需要以下两个命令：
      service portmap start
      注：在 新的内核中，NFS 守护进程改为 rpcbind，如是新内核，启动 NFS 守护进程的命令是“service rpcbind start”。 service nfs start
      到此 IP 为 192.168.1.2 的服务器已可以向其他两个节点提供 /usr/cluster 目录的文件共享。
2. 客户端配置方法（需要在所有子节点做同样的配置）。
   1) 建立共享目录。建立与服务器相同的共享目录用于共享服务器文件：
      mkdir /usr/cluster
   2) 查看服务器已有的共享目录。
   3) showmount -e 192.168.1.2

```
wjd@ubuntu:~$ showmount -e 10.201.8.195
The program 'showmount' is currently not installed. You can install it by typing:
sudo apt install nfs-common
wjd@ubuntu:~$ sudo apt install nfs-common
[sudo] password for wjd:
E: dpkg was interrupted, you must manually run 'sudo dpkg --configure -a' to correct th
e problem.
wjd@ubuntu:~$ sudo dpkg --configure -a
Processing triggers for mime-support (3.59ubuntu1) ...
Setting up tar (1.28-2.1ubuntu0.1) ...
Setting up man-db (2.7.5-1) ...
Building database of manual pages ...
wjd@ubuntu:~$ sudo apt install nfs-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  keyutils libnfsidmap2 libtirpc1 rpcbind
Suggested packages:
  open-iscsi watchdog
The following NEW packages will be installed:
  keyutils libnfsidmap2 libtirpc1 nfs-common rpcbind
0 upgraded, 5 newly installed, 0 to remove and 418 not upgraded.
Need to get 393 kB of archives.
After this operation, 1,370 kB of additional disk space will be used.
Do you want to continue? [Y/n]
Get:1 http://mirrors.tuna.tsinghua.edu.cn/ubuntu xenial/main i386 libnfsidmap2 i386 0.2
```

```
wjd@ubuntu:~$ showmount -e 10.201.8.195
Export list for 10.201.8.195:
/usr/cluster 10.201.8.206,10.201.8.199
wjd@ubuntu:~$
```

4) 通过这条命令我们可以查看 IP 地址为 192.168.1.2 服务器可以共享的目录情况。

5) 挂载共享目录。

mount -t nfs 192.168.1.2:/usr/cluster /usr/cluster

```
wjd@ubuntu:~$ mount -t nfs 10.201.8.195:/usr/cluster /usr/cluster
mount: only root can use "--types" option
wjd@ubuntu:~$ sudo -s
[sudo] password for wjd:
root@ubuntu:~# mount -t nfs 10.201.8.195:/usr/cluster /usr/cluster
mount.nfs: /usr/cluster is busy or already mounted
```

（第 2 次截图）

这一命令将 NFS 服务器 192.168.1.2 上的共享目录挂载到本地/usr/cluster 目录下。我们也可在所有子节点的/etc/fstab 文件中输入以下的代码，使文件系统在启动时实现自动挂载

NFS：

192.168.1.2:/usr/cluster /usr/cluster nfs defaults 0 0

至此我们已可以实现对 NFS 共享目录的本地访问，所有子节点的/usr/cluster 文件夹都共享了 NFS 服务器的同名文件夹的内容，我们可以像访问本地文件一样访问共享文件。MPICH 的安装目录和用户存放并行程序的文件夹都需要实现 NFS 共享，从而避免了每次向各节点发送程序副本。

(三) 配置 ssh 实现 MPI 节点间用户的无密码访问

由于 MPI 并行程序需要在各节点间进行信息传递，所以必须实现所有节点两两之间能

无密码访问。节点间的无密码访问是通过配置 ssh 公钥认证来实现的。例如，对新用户 user 配置 ssh 公钥认证，先在 node1 上做以下操作。

1) 生成了私钥 id_dsa 和公钥 id_dsa.pub，具体操作方法如下。

mkdir ~/.ssh

cd ~/.ssh

ssh-keygen -t dsa

```
wjd@ubuntu:~$ mkdir ~/.ssh
wjd@ubuntu:~$ cd ~/.ssh
wjd@ubuntu:~/.ssh$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/wjd/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/wjd/.ssh/id_dsa.
Your public key has been saved in /home/wjd/.ssh/id_dsa.pub.
The key fingerprint is:
SHA256:ike1bWZsBKgP6OyC5bDoGE19iwsX2bUOy7PychQgiy0 wjd@ubuntu
The key's randomart image is:
+---[DSA 1024]----+
|       ..        |
|  . . .  .       |
| o + o  o .      |
|E +.oo.o *       |
| +. +o=.S B      |
|.o+  Bo* =       |
|+*o +.B .        |
|*.oooo.o         |
|oo  .=o          |
+----[SHA256]-----+
wjd@ubuntu:~/.ssh$
```

2) 将该密钥用作认证，进行访问授权。按如下命令在 node1 执行。

cp ~/.ssh/id_dsa.pub ~/.ssh/authorized_keys chmod Go-rwx ~/.ssh/authorized_keys

3) 将~/.ssh 目录下的文件复制到所有节点。

scp -r ~/.ssh node2:

scp -r ~/.ssh node3:

```
wjd@ubuntu:~$ sudo -s
1qaz
[sudo] password for wjd:
Sorry, try again.
[sudo] password for wjd:
root@ubuntu:~# apt-get install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  ncurses-term openssh-client openssh-sftp-server ssh-import-id
Suggested packages:
  ssh-askpass libpam-ssh keychain monkeysphere rssh molly-guard
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
The following packages will be upgraded:
  openssh-client
```

```
root@ubuntu:~# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:7MyqD1VoWAP0EzE8Z3zsTB8fkxX89EljP263lkBK9JU root@ubuntu
The key's randomart image is:
+---[RSA 2048]----+
|    .oo*o .     .++|
|      +o=+ +.. +E.|
|     . =+.=...o++*|
|      . +  o..o.o+|
|       . S . o . .|
|      . +    . . o.|
|     .   +      o +|
|      . .         + |
|     .oo          .  |
+----[SHA256]-----+
```

4) 检查是否可以直接(不需要密码)登录其他节点。ssh node1 ssh node2

```
root@ubuntu:~# ssh-copy-id -i /root/.ssh/id_rsa node1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
The authenticity of host 'node1 (10.201.8.195)' can't be established.
ECDSA key fingerprint is SHA256:MmComyVfNUH3ppdet/k+DKMQkylhp3cGheU9KICxWQM.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any
 that are already installed

/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist on the
remote system.
                (if you think this is a mistake, you may want to use -f option)

root@ubuntu:~# ssh node1
Welcome to Ubuntu 14.04.2 LTS (GNU/Linux 3.13.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

New release '16.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Mon Jun 12 23:48:38 2017 from node2
root@node1:~# exit
注销
Connection to node1 closed.
```

```
root@ubuntu:~# ssh-copy-id -i /root/.ssh/id_rsa node2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any
 that are already installed

/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist on the
remote system.
                (if you think this is a mistake, you may want to use -f option)

root@ubuntu:~# ssh node2
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

320 packages can be updated.
129 updates are security updates.

Last login: Tue Jun 13 00:01:49 2017 from 10.201.8.195
root@ubuntu:~# exit
logout
Connection to node2 closed.
```

如能两两之间不需要密码登录其他节点，则表明配置成功。

(四) 安装 MPICH2

1. 下载并解压 MPICH2 压缩包。

   tar zxvf mpich2-1.3.2p1.tar.gz

解压完成后将在当前目录生成一个 MPICH 文件目录。

2. 进入 MPICH 解压后的目录，配置安装目录。

注意事项：

新版本 mpich2 的进程管理默认使用 hydra，而不是 mpd。如果你要使用 mpd，使用

./configure --with-pm=mpd:hydra

./configure --prefix=/usr/cluster/mpich2 --with-pm=mpd:hydra

```
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1$ sudo apt install gfortran
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  gfortran-5 libgfortran-5-dev
Suggested packages:
  gfortran-multilib gfortran-doc gfortran-5-multilib gfortran-5-doc
  libgfortran3-dbg
The following NEW packages will be installed:
  gfortran gfortran-5 libgfortran-5-dev
0 upgraded, 3 newly installed, 0 to remove and 418 not upgraded.
Need to get 8,359 kB of archives.
After this operation, 26.1 MB of additional disk space will be used.
```

```
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1$ ./configure --prefix=/usr/cluster/mpich2
--with-pm=mpd:hydra
```

```
config.status: creating testlist
config.status: creating cxx/testlist
config.status: creating f77/testlist
config.status: creating f90/testlist
config.status: creating threads/testlist
config.status: creating errors/testlist
config.status: creating errors/cxx/testlist
config.status: creating include/mpitestconf.h
config.status: executing default-1 commands
config.status: executing default-2 commands
config.status: executing default-3 commands
Configuration completed.
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1$
```

根据以上配置 MPICH 将安装在目录/usr/cluster/mpich2，并确保所有节点已建立针对该目录的 NFS 共享。

3. 编译安装 MPICH2。进入解压后的 MPICH 文件目录，分别执行 make 和 make install 指令，这会花一段较长的时间。

```
test -z "/home/wjd/Downloads/mpich2-1.3.2p1/lib/pkgconfig" || /bin/mkdir -p "/ho
me/wjd/Downloads/mpich2-1.3.2p1/lib/pkgconfig"
 /usr/bin/install -c -m 644 openpa.pc '/home/wjd/Downloads/mpich2-1.3.2p1/lib/pk
gconfig'
make[5]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1/src/openpa'
make[4]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1/src/openpa'
make[3]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1/src/openpa'
make[2]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1/src'
make[1]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1/src'
make all-postamble
make[1]: Entering directory '/home/wjd/Downloads/mpich2-1.3.2p1'
( cd examples && make )
make[2]: Entering directory '/home/wjd/Downloads/mpich2-1.3.2p1/examples'
  CC             cpi.c
../bin/mpicc   -o cpi cpi.o  -lm
make[2]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1/examples'
Make completed
make[1]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1'
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1$
```

```
make[3]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1/src/openpa'
make[2]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1/src/openpa'
make[1]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1/src/openpa'
make[1]: Entering directory '/home/wjd/Downloads/mpich2-1.3.2p1/src/pm/mpd'
copying python files/links into  /usr/cluster/mpich2/bin
make[1]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1/src/pm/mpd'
make[1]: Entering directory '/home/wjd/Downloads/mpich2-1.3.2p1/src/mpe2'
Installing MPE2 include files to /usr/cluster/mpich2/include
Installing MPE2 libraries to /usr/cluster/mpich2/lib
Installing MPE2 utility programs to /usr/cluster/mpich2/bin
Installing MPE2 configuration files to /usr/cluster/mpich2/etc
Installing MPE2 system utility programs to /usr/cluster/mpich2/sbin
Installing MPE2 man to /usr/cluster/mpich2/share/man
Installing MPE2 html to /usr/cluster/mpich2/share/doc/
Installed MPE2 in /usr/cluster/mpich2
/usr/cluster/mpich2/sbin/mpeuninstall may be used to remove the installation
make[1]: Leaving directory '/home/wjd/Downloads/mpich2-1.3.2p1/src/mpe2'
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1$
```

4. 在当前用户主目录下建立并编辑配置文件 mpd.hosts。将所有你允许访问本机进行并行计算的机器名填入，一行一个机器名，如果该机器上有两个CPU，就将它的名字加入两次，以此类推。

   node1

   node2

   node3

   node4

   注意，文中包含自己的目的是为了在只有一个节点时也可以模拟并行计算环境。

5. 配置环境变量。编辑用户主目录下的~/.bashrc 文件，增加一行：

   PATH="$PATH:/usr/cluster/mpich2/bin"

   这一行代码将 MPI 的安装路径加入用户的当前路径列表。重新打开命令行窗口后生效。

6. 启动 mpd 守护进程。运行 mpirun，首先要运行 mpd。在启动 mpd 守护进程前要在各个节点的安装目录（/usr/MPICHI-install/etc/mpd.conf）生成一个 mpd.conf 文件，内容为： secretword=123456

```
wjd@ubuntu:/etc$ which mpd
/usr/cluster/mpich2/bin/mpd
```

```
wjd@ubuntu:/etc$ which mpirun
/usr/cluster/mpich2/bin/mpirun
```

   其中，“123456”为识别口令，在所有节点中都建立该文件并保持口令一致，口令可自己设定。

   进入/usr/MPICHI-install/etc/目录执行以下命令

   touch mpd.conf

   chmod 600 mpd.conf

   mpd &

```
wjd@ubuntu:/usr/cluster/mpich2/etc$ sudo touch mpd.conf
wjd@ubuntu:/usr/cluster/mpich2/etc$
wjd@ubuntu:/usr/cluster/mpich2/etc$
wjd@ubuntu:/usr/cluster/mpich2/etc$
wjd@ubuntu:/usr/cluster/mpich2/etc$ sudo chmod 600 mpd.conf
wjd@ubuntu:/usr/cluster/mpich2/etc$
wjd@ubuntu:/usr/cluster/mpich2/etc$ sudo vi mpd.conf
wjd@ubuntu:/usr/cluster/mpich2/etc$ sudo gedit mpd.conf
```

```
wjd@ubuntu:~$ mpd&
[1] 14240
wjd@ubuntu:~$
```

mpd &为启动本地 mpd 的命令，我们也可以采用以下命令同时启动 mpd.hosts 中所列节点的 mpd。

mpdboot -n <节点个数> -f mpd.hosts

这一命令将同时在 mpd.hosts 文件中所指定的节点上启动 mpd 管理器。mpd 启动后执行“mpdtrace -l”可以查看各个节点机器名。

7. 编译、运行一个简单的测试程序 cpi，这是一个 MPICH 自带的计算圆周率的并行示例程序，该例程在 MPICH 解压后的 examples 目录下。
   运行命令如下：
   mpirun -np 3 ./cpi

```
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ mpirun -np 3 cpi
problem with execution of cpi  on  ubuntu:  [Errno 2] No such file or directory
problem with execution of cpi  on  ubuntu:  [Errno 2] No such file or directory
problem with execution of cpi  on  ubuntu:  [Errno 2] No such file or directory
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ mpirun -np 3 ./cpi
Process 0 of 3 is on ubuntu
Process 2 of 3 is on ubuntu
Process 1 of 3 is on ubuntu
pi is approximately 3.1415926544231323, Error is 0.0000000008333392
wall clock time = 0.000381
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ pm=mpd:hydra
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ pm=mpd:hydra
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ pm=mpd:hydra
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ pm=mpd:hydra
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ pm=mpd:hydra
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ pm=mpd:hydra
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ pm=mpd:hydra
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ pm=mpd:hydra
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ pm=mpd:hydra
wjd@ubuntu:~/Downloads/mpich2-1.3.2p1/examples$ pm=mpd:hydra
```

   mpi 的编译命令为 mpicc，如编译 test.c 可用如下命令：
   mpicc test.c -o test
   mpirun –np 3 ./test

从机部分（node3 部分截图，过程同上）

```
127.0.0.1       localhost
127.0.1.1       ubuntu

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff03::2 ip6-allrouters
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"/etc/hosts" 9 lines, 221 characters
```

```
cuijiawei@ubuntu:~/Desktop$ sudo -s
[sudo] password for cuijiawei:
root@ubuntu:~/Desktop# vi /etc/hosts
```

```
root@ubuntu:~/Desktop# apt-get install nfs-kernel-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
Some packages could not be installed. This may mean that you have
requested an impossible situation or if you are using the unstable
distribution that some required packages have not yet been created
or been moved out of Incoming.
The following information may help to resolve the situation:

The following packages have unmet dependencies:
 nfs-kernel-server : Depends: libtirpc1 but it is not going to be installed
                     Depends: nfs-common (= 1:1.2.8-6ubuntu1.2) but it is not go
ing to be installed
E: Unable to correct problems, you have held broken packages.
root@ubuntu:~/Desktop#
```

```
root@ubuntu:~/Desktop# apt-get install portmap
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'rpcbind' instead of 'portmap'
Some packages could not be installed. This may mean that you have
requested an impossible situation or if you are using the unstable
distribution that some required packages have not yet been created
or been moved out of Incoming.
The following information may help to resolve the situation:

The following packages have unmet dependencies:
 rpcbind : Depends: libtirpc1 but it is not going to be installed
E: Unable to correct problems, you have held broken packages.
```

```
127.0.0.1       localhost
127.0.1.1       ubuntu
10.201.8.195    node1
10.201.8.199    node2
10.201.8.206    node3
# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

```
gsharp@node1's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'gsharp@node1'"
and check to make sure that only the key(s) you wanted were added.

root@ubuntu:/home/cuijiawei/Documents# ssh gsharp@node1
Welcome to Ubuntu 14.04.2 LTS (GNU/Linux 3.13.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

1 package can be updated.
0 updates are security updates.

New release '16.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Mon Jun 12 21:12:22 2017 from node2
gsharp@node1:~$ exit
注销
Connection to node1 closed.
root@ubuntu:/home/cuijiawei/Documents# ssh-copy-id -i /root/.ssh/id_rsa.pub yfy@
node2
```

```
rtt min/avg/max/mdev = 1.794/9.964/161.701/14.992 ms

root@ubuntu:/home/cuijiawei/Documents#
root@ubuntu:/home/cuijiawei/Documents# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
13:74:b5:fa:7f:ee:54:89:7f:78:da:ee:54:2d:8c:7e root@ubuntu
```

```
The key's randomart image is:
+--[ RSA 2048]----+
|        . ...    |
|       . .   .   |
|        .   .    |
|         . . o. o|
|        S . ..o.+|
|         . o  .oo|
|            o E.=|
|             o *o|
|              o**|
+-----------------+
```

```
root@ubuntu:/home/cuijiawei/Documents# ssh-copy-id i /root/.ssh/id_rsa.pub gshar
p@node1
/usr/bin/ssh-copy-id: ERROR: Too many arguments.  Expecting a target hostname, g
ot: 'i' '/root/.ssh/id_rsa.pub' 'gsharp@node1'

Usage: /usr/bin/ssh-copy-id [-h|-?|-n] [-i [identity_file]] [-p port] [[-o <ssh
-o options>] ...] [user@]hostname
```

```
root@ubuntu:/home/cuijiawei/Documents# ssh-copy-id -i /root/.ssh/id_rsa.pub gsha
rp@node1
The authenticity of host 'node1 (10.201.8.195)' can't be established.
ECDSA key fingerprint is e9:ee:ce:07:7b:11:12:0e:85:55:55:31:5e:c6:d8:a9.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt
ed now it is to install the new keys
gsharp@node1's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'gsharp@node1'"
and check to make sure that only the key(s) you wanted were added.
```

```
root@ubuntu:/home/cuijiawei/Documents# ssh gsharp@node1
Welcome to Ubuntu 14.04.2 LTS (GNU/Linux 3.13.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

1 package can be updated.
0 updates are security updates.

New release '16.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Mon Jun 12 21:12:22 2017 from node2
gsharp@node1:~$ exit
注销
Connection to node1 closed.
```

```
root@ubuntu:/home/cuijiawei/Documents# ssh-copy-id -i /root/.ssh/id_rsa.pub yfy@
node2
The authenticity of host 'node2 (10.201.8.199)' can't be established.
ECDSA key fingerprint is bd:9c:4b:69:8a:a9:f8:e6:95:a6:12:33:39:95:f7:a8.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt
ed now it is to install the new keys
yfy@node2's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'yfy@node2'"
and check to make sure that only the key(s) you wanted were added.
```

```
root@ubuntu:/home/cuijiawei/Documents# ssh yfy@node2
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

320 packages can be updated.
129 updates are security updates.

Last login: Mon Jun 12 21:12:09 2017 from 10.201.8.199
yfy@ubuntu:~$ exit
logout
Connection to node2 closed.
root@ubuntu:/home/cuijiawei/Documents#
```

```
cuijiawei@ubuntu:~/Downloads/mpich2-1.3.2p1$ ./configure --prefix=/usr/cluster/m
pich2 --with-pm=mpd:hydra
```

```
config.status: creating errors/topo/Makefile
config.status: creating errors/io/Makefile
config.status: creating errors/cxx/Makefile
config.status: creating errors/cxx/errhan/Makefile
config.status: creating errors/cxx/io/Makefile
config.status: creating errors/f77/Makefile
config.status: creating errors/f90/Makefile
config.status: creating manual/Makefile
config.status: creating manual/manyconnect
config.status: creating perf/Makefile
config.status: creating testlist
config.status: creating cxx/testlist
config.status: creating f77/testlist
config.status: creating f90/testlist
config.status: creating threads/testlist
config.status: creating errors/testlist
config.status: creating errors/cxx/testlist
config.status: creating include/mpitestconf.h
config.status: executing default-1 commands
config.status: executing default-2 commands
config.status: executing default-3 commands
Configuration completed.
cuijiawei@ubuntu:~/Downloads/mpich2-1.3.2p1$
```

```
make[5]: Nothing to be done for 'install-exec-am'.
test -z "/home/cuijiawei/Downloads/mpich2-1.3.2p1/share/doc" || /bin/mkdir -p "/
home/cuijiawei/Downloads/mpich2-1.3.2p1/share/doc"
 /usr/bin/install -c -m 644 README '/home/cuijiawei/Downloads/mpich2-1.3.2p1/sha
re/doc'
test -z "/home/cuijiawei/Downloads/mpich2-1.3.2p1/lib/pkgconfig" || /bin/mkdir -
p "/home/cuijiawei/Downloads/mpich2-1.3.2p1/lib/pkgconfig"
 /usr/bin/install -c -m 644 openpa.pc '/home/cuijiawei/Downloads/mpich2-1.3.2p1/
lib/pkgconfig'
make[5]: Leaving directory '/home/cuijiawei/Downloads/mpich2-1.3.2p1/src/openpa'
make[4]: Leaving directory '/home/cuijiawei/Downloads/mpich2-1.3.2p1/src/openpa'
make[3]: Leaving directory '/home/cuijiawei/Downloads/mpich2-1.3.2p1/src/openpa'
make[2]: Leaving directory '/home/cuijiawei/Downloads/mpich2-1.3.2p1/src'
make[1]: Leaving directory '/home/cuijiawei/Downloads/mpich2-1.3.2p1/src'
make all-postamble
make[1]: Entering directory '/home/cuijiawei/Downloads/mpich2-1.3.2p1'
( cd examples && make )
make[2]: Entering directory '/home/cuijiawei/Downloads/mpich2-1.3.2p1/examples'
  CC              cpi.c
../bin/mpicc   -o cpi cpi.o  -lm
make[2]: Leaving directory '/home/cuijiawei/Downloads/mpich2-1.3.2p1/examples'
Make completed
make[1]: Leaving directory '/home/cuijiawei/Downloads/mpich2-1.3.2p1'
cuijiawei@ubuntu:~/Downloads/mpich2-1.3.2p1$
```

```
make[2]: Leaving directory '/home/cuijiawei/Downloads/mpich2-1.3.2p1/examples'
Make completed
make[1]: Leaving directory '/home/cuijiawei/Downloads/mpich2-1.3.2p1'
cuijiawei@ubuntu:~/Downloads/mpich2-1.3.2p1$ make install
```

主机部分（node 1）

(一) 准备系统和网络环境

局域网中三台安装有 64 位的 Ubuntu 的机器。

三台机器的 hostname 分别是 node1、node2 、node3 和 node4。我们将会用 node1 作为 MPICH2 环境中的 Server，node2 、node3 和 node4 作为 MPICH2 环境中的 worker。

Linux 系统中已经安装了 Python2.6、SSH 服务、NFS 服务。

步骤 1 配置 HOSTS 文件(前期阶段由于连接问题没有加入 node4，后面加上了 node4)

将修改/etc/hosts 文件实现 IP 解析，按如下格式进行添加

[root@node1 ~]# vi /etc/hosts

10.201.8.195 node1

10.201.8.199 node2

10.201.8.206 node3

hosts ×

```
127.0.0.1       localhost
127.0.1.1       unbuntu

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
furrent Version 17
# = prev + d(Wikipedia) + d(+Google)
# Newest Version http://go.netsh.org/link/getipv6hosts
127.0.0.1    localhost
::1    localhost
fe80::1%lo0    localhost
255.255.255.255    broadcasthost

10.201.8.195    node1
10.201.8.199    node2
10.201.8.206    node3

# Google Services inc G+ Gmail Drive …
# exc Youtube Videos

# g-Alternative-IP
# 2404:6800:4008:c02::11
# 2a00:1450:400b:c01::6a
# 2a00:1450:400c:c09::a0
# 64.233.189.118
```

步骤 2 配置 SSH

MPICH2 提供了一系列的命令对分布式计算环境中的各个进程(这些进程可能是在不同的机器中运行)进行管理，这些工具在后台都是用 SSH 来进行操作，所以将 Server 和 Worker 之间配置成无密码的 SSH 登陆可以方便后边的操作。

```
root@ubuntu:~# service ufw start
ufw start/running
root@ubuntu:~# service ufw stop
ufw stop/waiting
```

要求所有节点必须安装 openssh-server 而 ubuntu 默认不安装,故执行以下指令

[root@node1 ~]# apt-get install openssh-server ssh

```
root@ubuntu: ~
正在将新用户"statd" (UID 117)添加到组"nogroup"...
无法创建主目录"/var/lib/nfs"
statd start/running, process 14732
gssd stop/pre-start, process 14766
idmapd start/running, process 14818

正在处理用于 ureadahead (0.100.0-16) 的触发器 ...
正在设置 nfs-kernel-server (1:1.2.8-6ubuntu1) ...
Replacing config file /etc/exports with new version

Creating config file /etc/default/nfs-kernel-server with new version
 * Not starting NFS kernel daemon: no exports.
正在处理用于 libc-bin (2.19-0ubuntu6.6) 的触发器 ...
正在处理用于 ureadahead (0.100.0-16) 的触发器 ...
root@ubuntu:~# rpm -qa|grep nfs
root@ubuntu:~# apt-get install portmap
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
注意，选取 rpcbind 而非 portmap
rpcbind 已经是最新的版本了。
升级了 0 个软件包，新安装了 0 个软件包，要卸载 0 个软件包，有 1 个软件包未被升级
。
root@ubuntu:~# apt-get install nfs-kernel-server_
```

在 node1 上执行如下命令 生成密钥和公钥并分发至各个从机

[root@node1 ~]# ssh-keygen -t rsa

[root@node1 ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@node2

[root@node1 ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@node3

```
yfy@ubuntu: ~
Permission denied, please try again.
yfy@node2's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'yfy@node2'"
and check to make sure that only the key(s) you wanted were added.

root@ubuntu:~# ssh yfy@node2
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

320 packages can be updated.
129 updates are security updates.

Last login: Mon Jun 12 21:04:12 2017 from 10.201.8.195
yfy@ubuntu:~$ dir
1                  Documents       Pictures          simplescalar
1.c                Downloads       proc_clock.c      Templates
2.3                list            proc_clock.ko     test
app_sys_call       Makefile        proc_clock.mod.c  Videos
```

步骤 3 配置 NFS

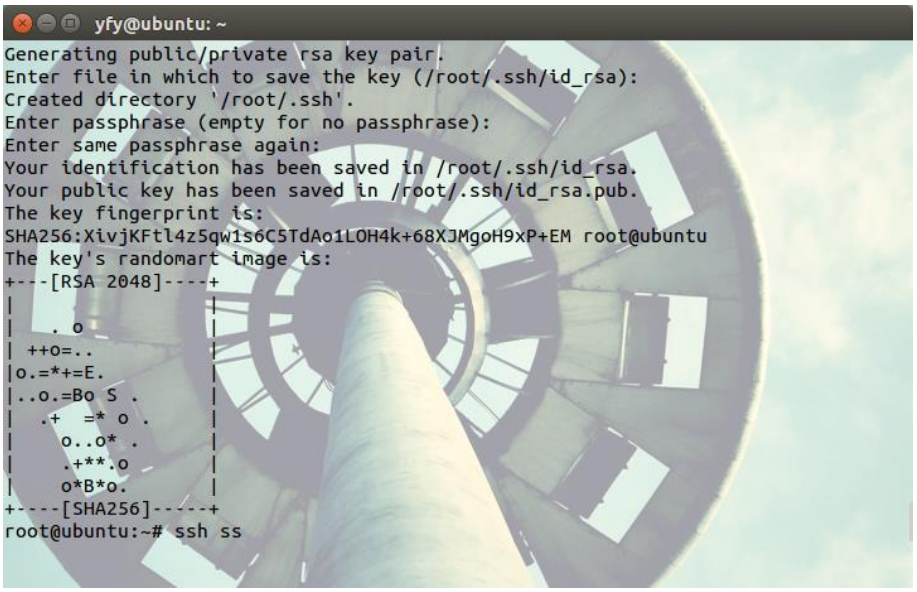
MPICH2 并没有提供一个统一的文件系统，所以需要借助操作系统提供的共享文件系统在 Server 和多个 Worker 之间共享数据。这里以 NFS 为例。

首先在 Server node1 上建立 NFS 共享目录：

Ubuntu 默认不配置 NFS，故需安装必要服务。

[root@node1 ~]# apt-get install rpcbind

[root@node1 ~]# apt-get install nfs-kernal-server




```
root@ubuntu:~# /etc/init.d/rpcbind restart
root@ubuntu:~# /etc/init.d/nfs-kernel-server restart
 * Stopping NFS kernel daemon                                    [ OK ]
 * Unexporting directories for NFS kernel daemon...              [ OK ]
 * Exporting directories for NFS kernel daemon...                [ OK ]
 * Starting NFS kernel daemon                                    [ OK ]
```

在 Server s1 上建立 NFS 共享目录：

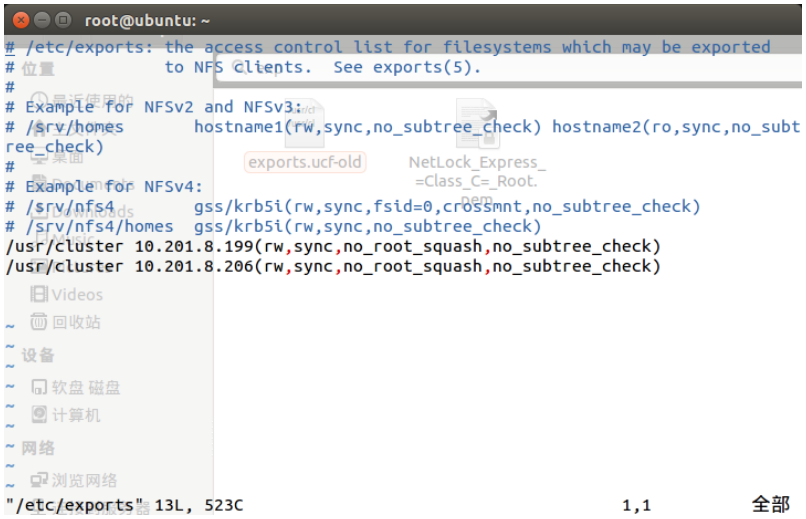
1. 创建共享目录：

   [root@node1 ~]# mkdir /usr/cluster

2. 将目录加入到 NFS 共享目录列表，在 /etc/exports 中添加如下配置：

   /usr/cluster node2(rw,sync,no_root_squash,no_subtree_check)

   /usr/cluster node3(rw,sync,no_root_squash,no_subtree_check)



3. 在 s1 上启动 NFS 服务：

   [root@node1 ~]# service nfs-kernel-server restart

4. 然后在 Worker s2 和 s3 上 mount s1 共享的 NFS 目录：

   [root@node2 ~]# mkdir /usr/cluster

   [root@node2 ~]# mount -t nfs node1:/usr/cluster /usr/cluster

   [root@node3 ~]# mkdir /usr/cluster

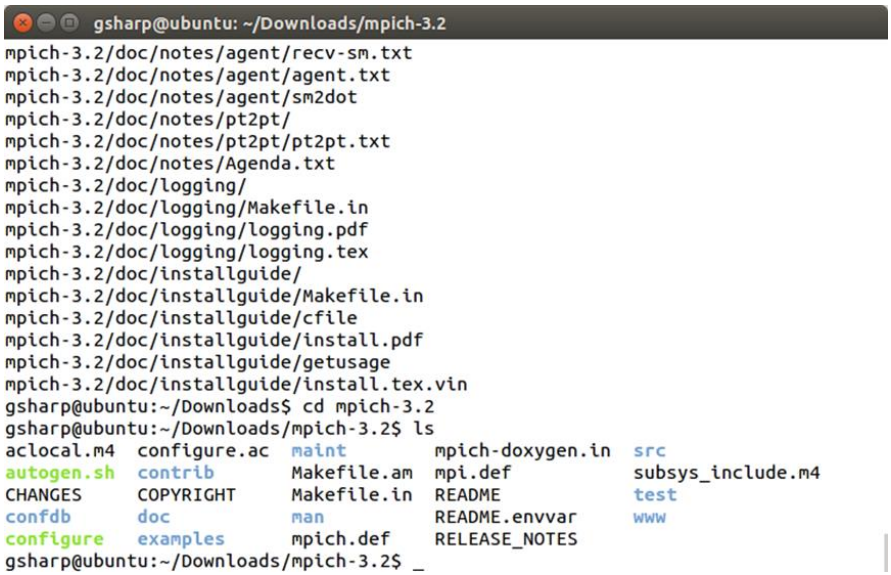
   [root@node3 ~]# mount -t nfs node1:/usr/cluster /usr/cluster

(二) 安装和配置 MPICH2

1. 下载并解压 MPICH2 压缩包。

   tar zxvf mpich2-1.3.2p1.tar.gz

   解压完成后将在当前目录生成一个 MPICH 文件目录。



2. 进入 MPICH 解压后的目录，配置安装目录。

   注意事项：

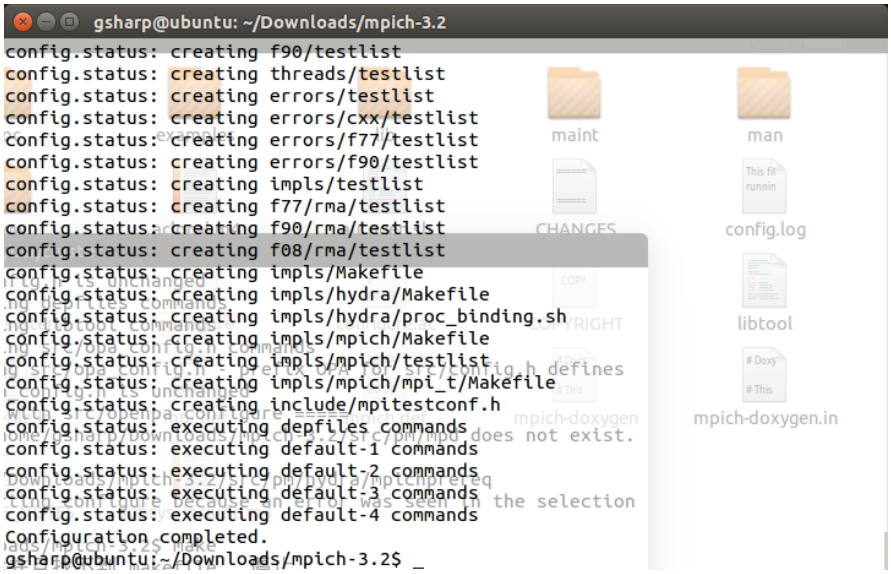
   最新版本 mpich2 的进程管理默认使用 hydra，而不是 mpd。如果你要使用 mpd，使用

   ./configure --with-pm=mpd:hydra

./configure --prefix=/usr/cluster/mpich2 --with-pm=mpd:hydra

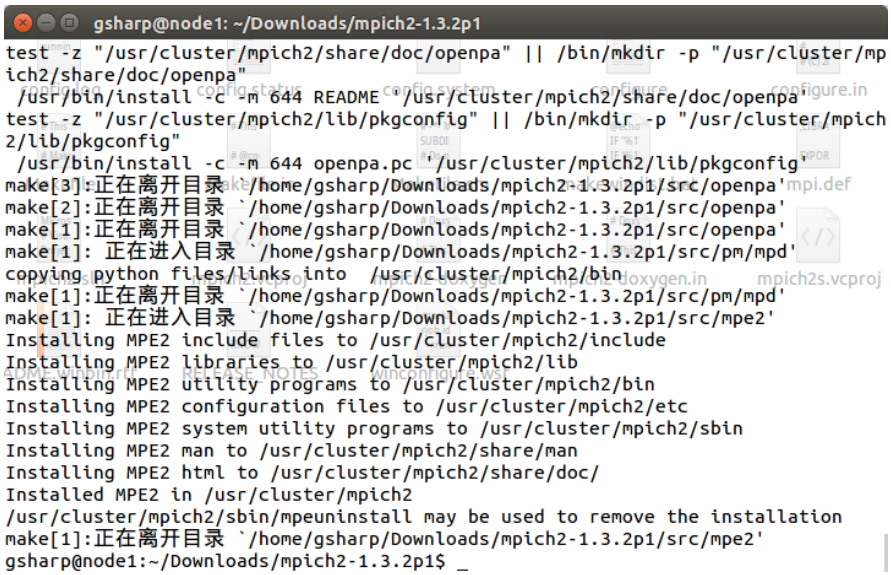
根据以上配置 MPICH 将安装在目录/usr/cluster/mpich2，并确保所有节点已建立针对该

目录的 NFS 共享。



3. 编译安装 MPICH2。

   进入解压后的 MPICH 文件目录，分别执行 make 和 make install 指令，这会花一段较长的时间。



4. 在当前用户主目录下建立并编辑配置文件 mpd.hosts。

   将所有你允许访问本机进行并行计算的机器名填入， 一行一个机器名，如果该机器上有两个 CPU， 就将它的名字加入两次，以此类推。

   node1

   node2

   node3

   node4

   注意，文中包含自己的目的是为了在只有一个节点时也可以模拟并行计算环境。

5. 配置环境变量。
   编辑用户主目录下的~/.bashrc 文件，增加一行：
   PATH="$PATH:/usr/cluster/mpich2/bin"
   这一行代码将 MPI 的安装路径加入用户的当前路径列表。重新打开命令行窗口后生效。
6. 启动 mpd 守护进程。
   运行 mpirun，首先要运行 mpd。在启动 mpd 守护进程前要在各个节点的安装目录（ /usr/MPICHI-install/etc/mpd.conf）生成一个 mpd.conf 文件，内容为：secretword=123456

四、 实验结果

并行计算

mpdboot -n <节点个数> -f mpd.hosts

这一命令将同时在 mpd.hosts 文件中所指定的节点上启动 mpd 管理器。

mpd 启动后执行“mpdtrace -l”可以查看各个节点机器名。

[root@node1 ~]# mpdtrace –l

编译、运行一个简单的测试程序 cpi，这是一个 MPICH 自带的计算圆周率的并行示例程序，该例程在 MPICH 解压后的 examples 目录下。

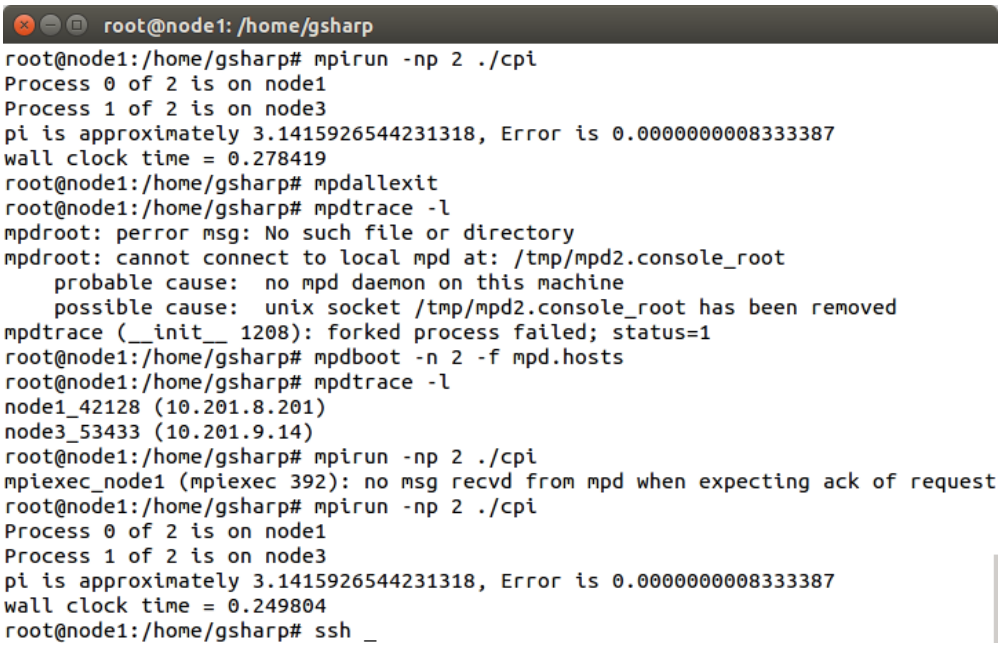
运行命令如下：

[root@node1 ~]# mpirun -np 3 ./cpi

mpi 的编译命令为 mpicc，如编译 hello.c 可用如下命令：

[root@node1 ~]# mpicc hello.c -o hello

[root@node1 ~]# mpirun –np 4 ./hello



```
root@node1: /home/gsharp
node2_47600 (10.201.9.7)
root@node1:/home/gsharp# mpdallexit
root@node1:/home/gsharp# mpdboot -n 2 -f mpd.hosts
root@node1:/home/gsharp# mpdtrace -l
node1_44142 (10.201.8.201)
node2_36922 (10.201.9.7)
root@node1:/home/gsharp# mpirun -np 2 ./cpi
Process 0 of 2 is on node1
Process 1 of 2 is on node2
pi is approximately 3.1415926544231318, Error is 0.0000000008333387
wall clock time = 0.225770
root@node1:/home/gsharp# mpdallexit
root@node1:/home/gsharp# mpdboot -n 3 -f mpd.hosts
root@node1:/home/gsharp# mpdtrace -l
node1_58972 (10.201.8.201)
node3_37207 (10.201.9.14)
node2_60634 (10.201.9.7)
root@node1:/home/gsharp# mpirun -np 3 ./cpi
Process 0 of 3 is on node1
Process 1 of 3 is on node3
Process 2 of 3 is on node2
pi is approximately 3.1415926544231318, Error is 0.0000000008333387
wall clock time = 0.143931
root@node1:/home/gsharp# _
```

```
root@node4: ~
root@node1:/usr/cluster# scp hello node3:/usr/cluster
hello                                         100% 8511     8.3KB/s   00:00
root@node1:/usr/cluster# mdpallexit
mdpallexit：未找到命令
root@node1:/usr/cluster# mpdallexit
root@node1:/usr/cluster# mpdboot -n 4 -f /home/gsharp/mpd.hosts
root@node1:/usr/cluster# mpdtrace -l
node1_37270 (10.201.8.201)
node4_41082 (10.201.9.8)
node2_33306 (10.201.9.7)
node3_51835 (10.201.9.14)
root@node1:/usr/cluster# mpirun -np 4 ./hello
helloworld!
helloworld!
helloworld!
helloworld!
root@node1:/usr/cluster# ssh node4
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

422 packages can be updated.
183 updates are security updates.

Last login: Tue Jun 13 00:09:57 2017 from 10.201.8.201
root@node4:~# _
```


root@node1: /usr/cluster
root@node1:/usr/cluster# mpirun -np 1 ./cpi
Process 0 of 1 is on node1
pi is approximately 3.1415926544231341, Error is 0.0000000008333410
wall clock time = 0.000924
root@node1:/usr/cluster# mpirun -np 4 ./cpi
Process 0 of 4 is on node1
Process 1 of 4 is on node4
Process 2 of 4 is on node3
Process 3 of 4 is on node2
^Croot@node1:/usr/cluster#
root@node1:/usr/cluster# mpirun -np 4 ./cpi
Process 0 of 4 is on node1
Process 1 of 4 is on node4
Process 2 of 4 is on node3
Process 3 of 4 is on node2
^Croot@node1:/usr/cluster#
root@node1:/usr/cluster# mpdallexit
root@node1:/usr/cluster# mpdboot -n 3 -f /home/gsharp/mpd.hosts
root@node1:/usr/cluster# mpirun -np 3 ./cpi
mpiexec_node1 (mpiexec 392): no msg recvd from mpd when expecting ack of reques
root@node1:/usr/cluster# mpirun -np 3 ./cpi
Process 0 of 3 is on node1
Process 1 of 3 is on node3
Process 2 of 3 is on node2
pi is approximately 3.1415926544231318, Error is 0.0000000008333387
wall clock time = 0.161189
root@node1:/usr/cluster# _
root@node1:/usr/cluster# mpdboot -n 4 -f /home/gsharp/mpd.hosts
root@node1:/usr/cluster# mpdtrace -l
node1_59709 (10.201.8.201)
node4_33666 (10.201.9.8)
node3_44669 (10.201.9.14)
node2_60619 (10.201.9.7)
root@node1:/usr/cluster# mpirun -np 4 ./hello
helloworld!
helloworld!
helloworld!
helloworld!
root@node1:/usr/cluster#