

1. MPI简介-环境搭建
2. MPI点对点通信
3. MPI集合通信
4. MPI 派生数据类型
5. MPI程序的性能评估
6. MPI实例：并行排序算法
7. 作业



太原理工大学
TAIYUAN UNIVERSITY OF TECHNOLOGY

MPI并行环境搭建

岳俊宏

E-mail:yuejunhong@tyut.edu.cn; Tel:18234095983



MPI环境搭建

搭建两台机器组成的一个集群；

采用linux操作系统，实现MPI并程序在两台电脑组成的集群上运行。





VMware-workstation-full-15.5.1-15018445.exe







CentOS-7-x86_64-Minimal-1810



MobaXterm

MobaXterm_Personal_10.5



	MobaXterm_Portable_v10.5	2020/2/17 14:17	文件夹
	VMware Workstation	2020/2/17 14:17	文件夹
	CentOS-7-x86_64-Minimal-1810.iso	2019/1/10 17:51	光盘映像文件
	mpich-3.0.4.tar.gz	2019/3/27 20:38	360压缩

- a) 安装虚拟机 (VMware)
- b) 安装CentOS7(注意打开网络、设置root密码123456、设置用户名hdusr, 密码123456);
- c) 安装必要的软件 (nano、net-tools、gcc、g++ gfortran)

```
sudo yum install nano
```

```
sudo yum install net-tools
```

```
sudo yum install gcc
```

```
sudo yum install gcc-c++
```

```
sudo yum install gcc-gfortran
```

```
////////////////////////////////////  
安装SSH服务：在每台机器上执行 sudo yum install ssh      (CentOS 7.x忽略此步)  
////////////////////////////////////
```

安装中可能出现的问题

虚拟机安装完成，装linux系统时，可能出现如下问题：

此主机支持Intel VT-x,但Intel VT-x处于禁用状态 异常解决；
可百度进行解决，如

<https://www.jianshu.com/p/74c02ad78bfc>

1. 进入BIOS进行如下设置，不同电脑可能会不同。
2. 选择Advanced--选择CPU Configuration--进入找到**Intel Virtualization Technology**--切换成**Enabled**--F10保存退出，等待计算机重启后以上异常即解决。

安装中可能出现的问题

Linux中"is not in the sudoers file"解决方法

当在终端执行sudo命令时，系统提示"hadoop is not in the sudoers file"：

其实就是没有权限进行sudo，解决方法如下（这里假设用户名是cuser）：

1.切换到超级用户：`$ su`

2.打开/etc/sudoers文件：`$vim /etc/sudoers`

3.修改文件内容：

找到"`root ALL=(ALL) ALL`"一行，在下面插入新的一行，内容是"`hadoop ALL=(ALL) ALL`"，然后在vim键入命令"`:wq!`"保存并退出。

注：这个文件是只读的，不加"`!`"保存会失败。

4.退出超级用户：`$ exit`

CentOS7版本后防火墙默认使用firewalld，因此在CentOS7中关闭防火墙使用以下命令：

1) 直接关闭防火墙

```
$ sudo systemctl stop firewalld.service
```

2) 禁止firewall开机启动

```
$ sudo systemctl disable firewalld.service
```

3) 关闭SELinux命令（永久关闭）

```
$ sudo nano /etc/selinux/config
```

注释下面两行：

```
#SELINUX=enforcing
```

```
SELINUX=disabled
```

```
#SELINUXTYPE=targeted
```

设置后需要重启才能生效。

```
$ sudo reboot -f
```

4) 查看防火墙状态：

```
$ /usr/sbin/sestatus
```

//////////这里以两台机器组成一个集群为例//////////

3.1) 首先克隆一台上述配置好的CentOS系统

3.2) 为每台机器配置静态IP地址：

- 设置IP 网关 DNS

查看获取的IP地址：\$ ip addr

若没分配IP地址-（输入命令dhclient，让CentOS 7自动获取一个IP地址：\$ dhclient）

- 所有相关的文件都位于目录/etc/sysconfig/network-scripts/下
- 打开配置文件：

\$ sudo nano /etc/sysconfig/network-scripts/ifcfg-ens33

修改内容如下：

```
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO="static"
DEERROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=ens33
#删除UUID,防止克隆时出现两台机器的唯一标识是一样的
DEVICE=ens33
ONBOOT=yes

#ip
IPADDR=192.168.190.145
#网关
GATEWAY=192.168.190.2
#子网掩码
NETMASK=255.255.255.0
#使用主的DNS
DNS1=192.168.190.2
#备用的DNS
DNS2=8.8.8.8
```

保存并关闭文件。

- 重启网络服务：

```
$ sudo systemctl restart network.service
```

3.3) 每台机器的主机名(hostname)改为唯一: Node1,Node2

修改机器名称:

第一台机器的机器名为Node1

第二台机器的机器名为Node2

在每台机器的终端窗口中, 执行命令:

```
$ sudo nano /etc/sysconfig/network
```

分别修改文件中的HOSTNAME为Node1和Node2

3.4) 设置每台机器的IP与主机名的对应关系

在/etc/hosts文件中存放的是域名与IP的对应关系;

```
$ sudo nano /etc/hosts
```

```
192.168.190.158 Node1
```

```
192.168.190.159 Node2
```

注：请将原文件最上面的第二行127.0.1.1 删除掉或者原内容全部删除(每台机器都要做)

4.1) 在每台机器上，执行如下命令：

```
$ ssh Node1
```

```
$ exit #记得最后通过这个命令退出ssh连接
```

```
$ cd .ssh
```

```
$ ssh-keygen -t rsa
```

然后一路回车，在.ssh下生成公私钥。

4.2) 在Node2上，使用如下命令将Node2的公钥发送给Node1：

```
$ scp ~/.ssh/id_rsa.pub hdusr@Node1:~/.ssh/1.pub
```

4.3) 在将Node2拷贝到Node1后，在Node1进行如下处理：

```
$ cp ~/.ssh/id_rsa.pub authorized_keys
```

```
$ cat ~/.ssh/1.pub >> authorized_keys
```

4.4) 在Node1上制作好的authorized_keys拷贝到Node2上:

执行如下命令:

```
$ scp ~/.ssh/authorized_keys hdusr@Node2:~/.ssh/
```

要想让这两台机器无密码登录, 需要更改一下必要的文件的权限 (所有节点都要更改): 其实只要保证authorized_keys的文件权限为600或者644

```
$ chmod 755 ~
```

```
$ chmod 755 ~/.ssh
```

```
$ chmod 600 ~/.ssh/authorized_keys
```

```
$ chmod 600 ~/.ssh/id_rsa
```

```
$ chmod 644 ~/.ssh/id_rsa.pub
```

4.5) 测试:分别在Node1和Node2上执行如下命令

```
$ ssh Node2
```

```
$ ssh Node1
```

这时会发现不需要输入密码, 直接就ssh连接上了这两台机器。

5.1) 在两台机器上的用户根目录下建立software目录，并把安装包拷到文件夹下，并对mpich-3.0.4.tar.gz进行解压：

```
$ tar zxvf mpich-3.0.4.tar.gz
```

5.2) 在Node1根目录下

```
$ mkdir mpi-install //安装MPICH的文件夹
```

```
$ mkdir mpi-work //并行政程序的文件夹
```

5.3) 进mpi解压包配置安装路径：

```
$ cd mpich-3.0.4
```

```
$ ./configure --prefix=/home/hdusr/mpi-install --with-pm=mpd:hydra
```

```
$ make
```

```
$ make install
```

5.4) 在用户主目录下建立文件夹mpich-install和mpi-work，并将其配置为NFS共享目录，即其路径为：/home/hdusr/mpi-install和/home/hdusr/mpi-work

/////共享程序文件夹，否则需要将程序拷贝到所有节点上/////

//////////5.4共享目录配置方法//////////

I. 在服务器端配置方法（在root权限下进行）：

a) /etc/exports文件配置，在文件/etc/exports中增加如下内容（以mpi-install为例）
/home/hdusr/mpi-install 192.168.202.0/24(rw,sync,no_root_squash)

b) 启动rpc和nfs服务

```
$ yum -y install nfs-utils//安装nfs
```

```
$ /bin/systemctl start rpcbind.service
```

```
$ /bin/systemctl start nfs.service
```

c) 设置开机自启动

```
$ systemctl enable rpcbind.service
```

```
$ systemctl enable nfs.service
```

II. 客户端配置方法（在用户权限下进行）：

```
$sudo yum -y install nfs-utils//安装nfs
```

a) 建立共享目录

```
$mkdir /home/hdusr/mpi-install
```

b) 查看共享目录

```
$showmount -e Node1(也可以写为Node1的IP地址)
```

c) 挂载共享目录

```
$sudo mount -t nfs Node1:/home/hdusr/mpi-install /home/hdusr/mpi-install
```

```
$ df -h//查看是否挂载
```

d) 客户端的开启自动挂载设置(推荐)

```
$sudo nano /etc/fstab //在该文件中增加如下内容
```

```
Node1:/home/hdusr/mpi-install    /home/hdusr/mpi-install    defaults    0    0    17
```

5.5) 安装后分别在Node1和Node2加入环境变量/etc/profile (~/.bashrc)

\$sudo nano ~/.bashrc 或者 \$ sudo nano /etc/profile

PATH=\$PATH:/home/hdusr/mpi-install/bin

MANPATH=\$MANPATH:/home/hdusr/mpi-install/man

export PATH MANPATH

\$source ~/.bashrc 或者 \$ source /etc/profile

// 检测运行的mpi命令的版本:

\$ which mpicc

\$ which mpiexec

5.6) 设置、启动mpd守护进程

a) 在用户主目录下创建文件 ~/mpd.hosts

将所有允许访问本机进行并行计算的机器名填入，一个机器名一行。

在本实例中的~/mpd.hosts文件内容为：

Node1

Node2

b) 设置进程的配置环境

MPI通过mpd管理器来管理运行MPI程序。此外，需要在用户主目录下为其创建进程管理的配置文件（基于安全考虑）：.mpd.conf，

注意：这个文件只能由用户读写，在每台机器上创建如下文件：

```
$cd /home/hdusr
```

```
$touch .mpd.conf
```

```
$sudo nano .mpd.conf
```

```
secretword=123456//必须一致
```

```
$chmod 600 .mpd.conf
```

```
$mpd& //启动本地mpd的命令
```

```
$ mpdboot -n 2 -f ~/mpd.hosts
```

```
$ mpicc -g -Wall -std=c99 -o mpi_hello mpi_hello.c
```

```
$ mpiexec -n 5 ./mpi_hello
```

```
$ mpirun -n 5 ./mpi_hello
```

```
$ mptest 1000 // 消息传递测试
```

- ❑ MPI进程的创建、启动和管理需借助进程管理（PM）来完成。
- ❑ PM就是MPI环境与操作系统的接口
- ❑ MPICH提供多种进程管理器，如MPD
- ❑ MPD是由Python实现的一组工具构成。
- ❑ Mpdboot在多个主机上启动MPD进程，形成MPI运行时的环境，目标主机列表由\$HOME/mpd.hosts文件制定

//同时启动mpd.hosts中所列节点的mpd

\$ mpdboot -n 3 -f mpd.hosts

\$ mpdtrace -l //查看节点列表

\$ mpdcleanup//关闭

\$ mpdexit //杀死指定节点mpd守护进程。

常用参数: -mpdid

\$ mpdallexit //杀死所有的mpd守护进程。

\$ mpdlistjobs //列出作业的进程信息。

\$ mpdkilljob //停止某个作业的所有进程

```
$ mpiexec -n <number of processes> <executable>
```

其中-n是表示用几个进程来执行这个程序。

MPI-2标准建议使用mpiexec代替mpirun来启动应用程序。

MPICH2实现了mpiexec标准并通过MPD对标准做了适当的扩展。

mpiexec提供MPI环境与外界交互的强大接口，并在此基础上构造MPI进程管理环境。

- ❑ `mpiexec -n 1 -host loginnode master: -n 32 -host slave`

其作用范围为由冒号隔开的选项组

- ❑ `mpiexec -gdb -n 10 ./cpi`

在启动之前需要使用-g选项来编译程序

启动程序后，各进程均分别在各自环境和GDB控制下执行。

`mpdlistjobs`:查看各节点上运行的进程信息

- ❑ `totalview`是一个商业版本的图形化调试工具。

1. 完成具有两个节点的集群搭建，并进行测试。



结束！

设置开机自启动

systemctl enable rpcbind.service

systemctl enable nfs.service

查看是否开机自启动

systemctl list-unit-files |grep enabled

|grep: 目的是过滤，列出所有的enabled的情况

systemctl list-unit-files rpcbind.service

systemctl list-unit-files nfs.service

查看状态

systemctl status rpcbind.service

systemctl status nfs.service

- 查看服务器启动状态

```
[root@Node1 hdusr]# systemctl status rpcbind.service
● rpcbind.service - RPC bind service
   Loaded: loaded (/usr/lib/systemd/system/rpcbind.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2020-02-24 01:35:04 EST; 7min ago
     Process: 5896 ExecStart=/sbin/rpcbind -w $RPCBIND_ARGS (code=exited, status=0/SUCCESS)
    Main PID: 5988 (rpcbind)
      CGroup: /system.slice/rpcbind.service
              └─5988 /sbin/rpcbind -w

Feb 24 01:35:04 localhost.localdomain systemd[1]: Starting RPC bind service...
Feb 24 01:35:04 localhost.localdomain systemd[1]: Started RPC bind service.
```

在服务器端配置方法中，root权限下修改/etc/exports文件；

如果期间有修改共享文件夹目录，则需要用以下命令刷新以下：

exportfs -rv