講義 1

ディスクフルクラスタの構築

同志社大学 大学院　工学研究科
大西 祥代
onishi@mikilab.doshisha.ac.jp

■　はじめに

シミュレーションや解析の分野では，高性能な計算資源が必要で，そこで注目されているのが PC クラスタである．PC クラスタは複数の汎用 PC を接続し，仮想的に一つの計算機とする技術である．PC クラスタには全てのノードがディスクを持つディスクフルクラスタと，ディスクを持たないディスクレスクラスタとがあり，その構築方法は異なる．本報告では，ディスクフルクラスタの構築について述べる．

■　ディスクフルクラスタとは

ディスクフルクラスタとは, 計算ノードがディスクフルマシンの PC クラスタである．ディスクフルマシンとは, ディスクを使用するマシンのことである．ディスクフルマシンは， 図 1.1 のようにマスターノード (master) と計算ノード (slave1, slave2…) によって構成されており, 各計算ノードがハードディスクを持っている．マスターノードと計算を行うマシンは, 内部ネットワークで繋がっており, マスターノードのみが外部ネットワークと繋がっている．各マシンは, まずカーネルが起動し, ハードディスクにあるルートファイルシステムにマウントする．そして, いくつかの起動スクリプトを読み込み実行する．この起動スクリプトの一つに, ネットワークに関する設定を行うスクリプトが含まれており, これにより各マシンがネットワーク接続する．

■　構築手順

□　OS のインストール

クラスタに用いる全てのマシンに Debian をインストールする．ここでは，Debian のインストールはすでに終わっているものとする．

□　ネットワークの設定

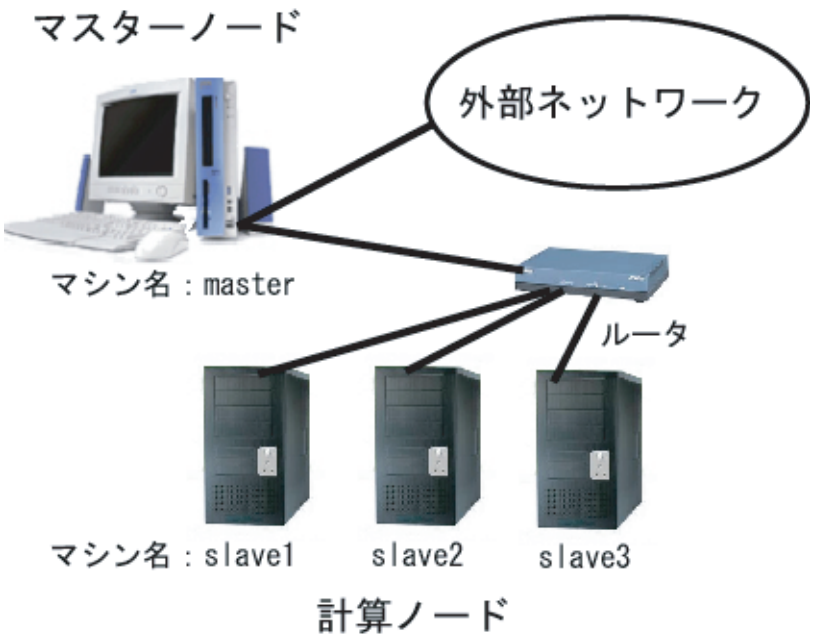

図 1.1: クラスタの構成

ネットワークインターフェースに関するアドレス等の設定は以下で行う.

```
# vi /etc/network/interfaces
```

マスターノードでのネットワークの設定

マスターノードでは, 外部との通信・計算ノードとの通信という 2 つのネットワークインターフェイスの設定を行う．本報告では，eth0 を外部用, eth1 を内部用に設定する．eth0 は, DHCP で設定しても, 固定 IP アドレスを割り当ててもどちらでもよい.

- eth0 を DHCP により自動的にアドレスを割り当てる場合

```
auto eth0
iface eth0 inet dhcp
```

- eth0 に固定 IP アドレスを割り当てる場合の例

```
auto eth0
iface eth0 inet static
address 172.20.11.200
network 172.20.11.0
netmask 255.255.255.0
broadcast 172.20.11.255
gateway 172.20.11.1
```

計算ノードでのネットワークの設定

計算ノードでは，内部での通信の設定のみを行う．

eth1 は以下のように IP アドレスを設定する．

- eth1 を設定する場合の例

```
auto eth1
iface eth1 inet static
address 192.168.1.1
network 192.168.1.0
netmask 255.255.255.0
broadcast 192.168.1.255
```

networking の再起動

ネットワークの設定を反映するために networking を再起動する．

```
# /etc/init.d/networking restart
```

マスターノードマシンと計算ノードのネットワークの IP アドレスの設定は以下のように行う．

ホスト名	外部アドレス	内部アドレス
master	172.20.11.200	192.168.1.1
slave1	-	192.168.1.2
slave2	-	192.168.1.3
slave3	-	192.168.1.4

□ カーネルの再構築

カーネル 2.6 では, NFS(Network File System) がサポートされているので, カーネルの再構築の必要はない. サポートされていない場合には, 次のように設定を行う.

カーネル再構築する際に必要なパッケージと，カーネルソースをインストールする．

```
# aptitude install libncurses5-dev     (make menuconfig に必要)
# aptitude install kernel-source     (バージョンがいくつかあるが kernel-source-2.6.8 を使用する)
# aptitude install kernel-package     (make-kpkg コマンドに必要)
```

kernel-source-2.6.8 を解凍する．

```
$ tar jxvf kernel-source-2.6.8.tar.bz2
```

カーネルソースがおかれるディレクトリに移動する．

```
# cd /usr/src/kernel-source-2.6.8
```

次のコマンドにより, 設定画面が出てくる．

```
#make menuconfig
```

- マスターノードの設定

```
Filesystem を選ぶ- - - -
Network File Systems を選ぶ- - - -
[*] NFS server support
(シフトキーで「*」を選び「組み込み」にする.)
[*] Provide NFSv3 server support
```

- 計算ノードの設定

```
Filesystem を選ぶ- - - -
Network File Systems を選ぶ- - - -
[*] NFS server support
[*] Provide NFSv3 server support
```

NFS に関する設定を行った後，コンパイルを行う.

```
# make dep (インクルードファイルなどの依存関係が正しいかを確認する)
# make-kpkg clean (不要ファイルを削除する)
# make-kpkg –revision 20070406 kernel-image (dep パッケージが作られる)
```

コンパイルに成功すれば，インストールを行う.

```
# cd /usr/src
# dpkg -i kernel-image-2.6.8_20070406_i386.deb
```

再起動する.

```
# reboot
```

□ rsh

rsh(remote shell) とは, リモートシステム上で指定したコマンドを実行するためのコマンドである.

マスターノードでの rsh インストールと設定

rsh クライアントをインストールする. セキュリティの問題上, マスターノードとなるノードには rsh サーバはインストールせず，クライアントのみをインストールする.

```
# aptitude install rsh-client
```

DNS を用いないクラスタでは, IP アドレス, ドメイン名およびホスト名の照合を行う.

/etc/hosts の編集

IP アドレスと hostname の編集を行う．下記のように IP アドレスとホスト名を記述する．ドメイン名の部分，つまり master.domain.name や slave1.domain.name などは書かなくてもよい.

```
127.0.0.1 localhost.localdomain hostname
#IP address domainname hostname
192.168.1.1     master.domain.name     master(ホスト名を書く)
192.168.1.2     slave1.domain.name     slave1(ホスト名を書く)
192.168.1.3     slave2.domain.name     slave2(ホスト名を書く)
192.168.1.4     slave3.domain.name     slave3(ホスト名を書く)
```

マシンのホスト名は

```
# hostname
```

と入力すると確認できる．

計算ノードでの rsh インストールと設定

計算ノードには rsh サーバと rsh クライアントをインストールする．

```
# aptitude install rsh-client rsh-server
```

/etc/hosts の編集

IP アドレスと hostname の編集を行う．下記のように IP アドレスとホスト名を記述する．ドメイン名の部分，つまり master.domain.name や slave1.domain.name などは書かなくてもよい．

```
127.0.0.1 localhost.localdomain hostname
#IP address domainname hostname
192.168.1.1     master.domain.name     master(ホスト名を書く)
192.168.1.2     slave1.domain.name     slave1(ホスト名を書く)
192.168.1.3     slave2.domain.name     slave2(ホスト名を書く)
192.168.1.4     slave3.domain.name     slave3(ホスト名を書く)
```

クラスタを構成しているノード間の通信は, パスワードなしで行う．そこで, パスワードなしを許可するノードの IP アドレスまたはホスト名を登録する．

/etc/hosts.equiv の編集

```
192.168.1.1(ホスト名でもよい)
192.168.1.2(ホスト名でもよい)
192.168.1.3(ホスト名でもよい)
192.168.1.4(ホスト名でもよい)
```

設定変更を有効にするために，inetd を再起動する．

```
# /etc/init.d/inetd restart
```

/etc/init.d/inetd がない場合は openbsd-inetd を再起動する．

```
# /etc/init.d/openbsd-inetd restart
```

root 権限で rsh を使用する場合は，下記のように PAM モジュールを書き換えることでパスワードを聞かれなくなる．下記のようにコメントアウトや書き加えを行う．

/etc/pam.d/rsh の編集

```
auth required pam_nologin.so
#auth required pam_securetty.so
auth required pam_env.so
#auth required pam_rhosts_auth.so
auth required pam_rhosts_auth.so  hosts_equiv_rootok promiscuous
account required pam_unix_acct.so
session required pam_unix_session.so
```

/etc/pam.d/rlogin の編集

```
auth required pam_nologin.so
#auth required pam_securetty.so
#auth sufficient pam_rhosts_auth.so
auth sufficient pam_rhosts_auth.so  hosts_equiv_ rootok promiscuous
auth required pam_env.so
#auth required pam_rhosts_auth.so
auth required pam_unix.so nullok
account required pam_unix.so
password required pam_unix.so nullok use_authtok obscure
min=4 max=8
```

□ MPI

クラスタでは, マシン間でメッセージの授受が必要で, MPI(Message Passing Interface) によりそれを行う. MPI とは並列処理アプリケーション用メッセージパッシングライブラリである. 今回使用する MPICH は MPI の実装の一つである.

MPICH のインストール

マスターノード, 計算ノードに MPICH をインストールする.

```
# aptitude install mpich
```

MPICH の設定

MPICH は, 計算ノードのみ設定を行う. 並列計算に用いるノードの IP アドレスまたはホスト名を登録する. ここに登録されていないノードは使用されない. マスターノードは計算しないので, 登録しない.

/etc/mpich/machines.LINUX の編集

```
#IP address
192.168.1.2(ホスト名でもよい)
192.168.1.3(ホスト名でもよい)
192.168.1.4(ホスト名でもよい)
```

□ NFS

並列計算を行う際には, 実行ファイルやそれに用いられるソフトウェアを全ノードが持っていなければならない. NFS(Network File Systems) を用いると, NFS クライアントは NFS サーバのディスクにアクセスでき, ファイル共有が可能となる.

マスターノードの NFS のインストールと設定

マスターノードが NFS サーバになる. NFS サーバをインストールする.

```
# aptitude install nfs-kernel-server
```

/etc/exports の編集

slave1，slave2，slave3 は各計算ノードのホスト名である.

```
/home slave1(rw,sync) slave2(rw,sync) slave3(rw,sync)
```

NFS サーバを再起動する.

```
# /etc/init.d/nfs-kernel-server restart
```

計算ノードの NFS のインストールと設定

計算ノードが NFS クライアントとなる.

```
# aptitude install nfs-common
```

/etc/fstab の編集

起動時に自動的に NFS サーバにマウントするように設定する.

```
マスターノードのホスト名:/home /home nfs defaults,rw 0 0
```

NFS クライアントを再起動するには下記のコマンドを実行する．ここでは再起動の必要はない.

```
# /etc/init.d/nfs-common restart
```

NFS サーバのディスクにアクセスするために以下のコマンドでマウントする.

```
# mount -a
```

□ NIS

NIS(Network Information Service) とは, ネットワーク上で全ての計算機で必要な情報を共有するサービスのことである. ユーザ名が同じでも使うマシンが違うと ID は一致せず, 別のユーザと認識されてしまう. そこで, NIS を使い, 異なるノード間でも同じユーザ ID, グループ ID が使用できるようにしている. マスターノードマシンが NIS サーバに, 計算ノードが NIS クライアントとなる.

マスターノードの NIS のインストールと設定

マスターノードに NIS をインストールする．インストール中に, NIS ドメイン名の入力を求められるので, 任意で入力する．セキュリティ上 DNS サーバとは違うものにする．

```
# aptitude install nis
```

次に, マスターノードが NIS サーバとなるよう設定を行う．

/etc/init.d/nis の編集

```
NISSERVER=master
```

/etc/default/nis の編集

```
NISSERVER=master
```

NIS サーバが提供する情報 (ユーザ, グループ) はマップと呼ばれる．下のコマンドでマップの作成を行う．

```
# /usr/lib/yp/ypinit -m
```

NIS サーバ名を聞かれるので入力し, 追加を行わないので Ctrl+D を押す．

新しくユーザを追加する場合はマスターノードのみの設定でよく, 追加した後は以下のコマンドを実行する．

```
# cd /var/yp
# make
```

□ 計算ノードの NIS のインストールと設定

計算ノードに NIS をインストールする．

```
# aptitude install nis
```

計算ノードが NIS クライアントとなるよう設定を行う．

/etc/init.d/nis の編集

```
NISSERVER=false
```

etc/default/nis の編集

```
NISSERVER=false
```

NIS サーバを設定する．

/etc/yp.conf の編集

```
ypserver マスターノードのホスト名
```

/etc/passwd と/etc/group の編集

最後の行に追加する．

```
+::::::
```

/etc/nsswitch.conf の編集

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

NIS の再起動

設定を反映するために NIS を再起動する.

```
# /etc/init.d/nis restart
```

□ テスト

まず自分のホームディレクトリに計算させるマシンのリストファイルを作成する.

```
$ cd /home/onishi
$ vi machinelist
```

ファイルは以下のように計算させるマシンのホスト名を記述する.

```
slave1
slave2
slave3
```

次にサンプルプログラムをコピーし, コンパイルして実行ファイルを生成した後に並列計算を行う.

```
$ cp /usr/share/doc/libmpich1.0-dev/examples/cpi.c /home/onishi(自分のホームディレクトリ)
$ mpicc cpi.c
$ mpirun -np プロセッサ数 -machinefile machinelist a.out
```

計算に用いたマシン名と円周率の計算結果が表示されていることを確認する.

■ エラー対処

計算に用いたマシン名，または計算結果が表示されない場合の対処法について述べる.

- rsh の確認

  以下のコマンドで，計算ノードと接続できているか確認する.

  ```
  # rsh ホスト名
  ```

  これが失敗した場合，rsh の設定に問題があると考えられる.

- マウント状況の確認

  計算ノードで以下のコマンドを入力し，マスターノードと情報共有ができているか確認する.

```
# df
```

マスターノードと情報共有できていない場合は NFS の設定に問題があると考えられる.