ディスクレスノード構築のためのメモ

牧野淳一郎

December 29, 2004

Contents

1 前置き **3**

2 SYSLINUX **3**

3 ルートファイルシステムの作成 **5**

4 syslinix.cfg の設定 **7**

5 shutdown sequence の変更。 **7**

6 マシン毎の設定について **8**

6.1 USB メモリ 8

6.2 ルートファイルシステム 9

6.3 マザーボード BIOS の設定。 10

7 問題点と解決方法 **10**

7.1 なにも考えないでリセットすると次のブートでこける 10

1 前置き

GRAPE-6 のホストやらなにやらで計算機が何十台もあるので、いくらかでも管理を簡単にするために diskless のシステムを構築することを考えた。

やり方はいろいろあるし、出来合いのパッケージみたいなのも一杯あるみたいなんだけど、 そういうのは使わないでなるべく基本的なものだけを使うことにする。

その理由は:

1. ディスクレスクラスタがどういうふうに動くかのそもそもの仕組みを理解したい。
2. 何かうまくいかなかった時に解決できるようにしたい。

というようなのがたてまえ。本当の理由の少なくとも一部は、「そのほうが面白いから」である。

で、色々資料とかを読んで調べると、どうも、以下のような方法が普通そうであることがわかる。

1. FD または CD からブート。
2. PXELinux とかそういうものを使ってネットワークブート

いずれの場合でも、ルートファイルシステムは NFS 先にあることになる。ルートとしてマウント出来さえすれば、後は普通に HDD からブートするのと特に変わらないように見える。特に、もっとも単純にはディスクレスノード毎に完全なルートファイルシステムを準備してしまえば、管理の手間はともかくあまり難しいことなくシステムが組めそうである。

だが、この 2 つともなんとなく気にいらない。 FD, CD からのブートは、そもそもそういうものをつけないといけないのが面倒だし、ノード数だけメディアを準備するのもやはり面倒である。 といって、ネットワークブートは、 NFS サーバーの他に DHCP とか色々設定しないといけないものが増えるので面倒である。

ブートメディアとしてもうちょっと手軽なものはないかと考えてみると、もちろん USB メモリというものがある。USB メモリからブート出来る Linux なんてものもいろいろあるので、 USB メモリからブートしてディスクレスというのも悪くはなさそうである。

上手く出来ればこれはいろいろ利点がある。

1. フロッピーからブートするのとほとんど手順は同じで、最初のとっつきが簡単。
2. マザーボードにコネクタがあるのでドライブをつけなくても使えて、フロッピー、CD に比べて手軽である。また、メディアの書き変えも早い。カーネル、ブートローダをコピーするのが数秒で終わる。
3. メディアの値段もそんなに高くない。

というわけで、 USB メモリからカーネルをロード、ブートして、ルートファイルシステムは NFS という (微妙に中途半端な) システムを構築することにする。

なお、以下の記述をみてディスクレスクラスタを作ろうという人 (いるのか？) に注意。これは TurboLinux8 for AMD64 に特定の記述が一杯あると思うので、違うディストリビューションの場合には違う設定・作業が必要になると思います。

2 SYSLINUX

さて、 USB メモリからブートするには何かブートローダがいる。 Linux のブートローダといえば LILO か GRUB だが、 USB メモリから ブートする Linux とかは大抵 SYSLINUX を使ってるのと、これはネットワークブート用の PXELINUX の親戚らしいのでこれを使ってみることにする。

とりあえず、ものをどこかから持ってくる。配布元は

http://syslinux.zytor.com/

のようである。といっても、 download のリンクをたどると:

http://www.kernel.org/pub/linux/utils/boot/syslinux/

に。新しそうなバージョンの syslinux-x.xx.tar.gz をもらってくる。で、展開するとなんか一杯ファイルがでてくるが、本当にいるのは実行バイナリである syslinux だけ。これは基本的には単に

```
syslinux /dev/sda1
```

とすると (USB メモリが SCSI 互換モードで /dev/sda として認識されているとして)、ブートセクタになんか書くというものである (らしい)。この時、USB メモリ (というか、本来フロッピー用のブートローダらしいけど) のファイルシステムは FAT16 でないといけなくて、そこの syslinux.cfg という設定ファイルを読んでカーネルをロードしたりパラメータを設定したりしてくれる。

とりあえず、なにかカーネルをいれてブートするかどうかを試してみる。

FAT16 でないといけないが、最近の USB メモリは FAT32 であるのでちょっと面倒なこともある。 Windows からなら format f: /fs:fat (f は USB に対応するドライブレターに変える) でいいけど、 Linux からだと、、、とりあえず、私が生協で買ったものについては、以下のようなスクリプトで USB メモリの必要な設定が全てできる。

```
/sbin/sfdisk --force /dev/sda << EOF
0,1014,6,*
EOF
/sbin/fdisk -l /dev/sda
/sbin/mkfs.msdos /dev/sda1
./syslinux /dev/sda1
mcopy syslinux.cfg c:
mcopy vmlinuz c:
mdir c:
```

ここで sfdisk して mkfs.msdos してるのは、なんか Linux から見えない妙なパーテッションテーブルがあったのを上書きするため。こういう時には sfdisk は素晴らしく便利である。

mcopy は /etc/mtools.conf で

```
drive c: file="/dev/sda1"
```

.mtoolsrc で

```
mtools_skip_check=1
```

をして無理矢理書くようにしている。後者は多分必要ないかも。まあ、mtools を使わなくても、 mount してコピーすればすむという話もある。ここで、使うカーネルはとりあえず適当なもの。ブートするかどうか見るだけだから。設定ファイルのほうは

```
DEFAULT vmlinuz
TIMEOUT 3
PROMPT 1
```

とか書いてあればなんかすると思う。で、 USB メモリを PC に挿してブート。この時、 BIOS の設定で HDD とかより FDD を優先するようにしておくと、 USB も FDD だと思ってそこからブートしてくれるものが多いと思うけど、これはマザーボードによる。

さて、普通はここで Linux とか出るけどそこから先にいかない、とかいうことになる。これは、大抵はファイルシステムが実は FAT32 だったりするせいなので、確認してみること。

これでブートしたとして、もちろんルートファイルシステムがないし、カーネルのほうも NFS root mount ができる設定になってないので、カーネルがひとしきりハードウェアの認識とかしておもむろに root をマウントしたいところでハングする。とはいえ、ここまでこればディスクレスノードの作成は出来たも同然である。

3 ルートファイルシステムの作成

ファイルサーバにする機械とディスクレスノードで、基本的には同じ OS を動かすことにする。で、サーバにする機械で、ディスクレスノード用のものを置くところを

```
/home2/clients
```

ということにして、まず、なにも考えないで

```
tar cvf /home2/clients/root.tar -X  /home2/clients/excludes /
```

で全体ファイルを作成、

```
mkdir /home2/clients/test1
cd /home2/clients/test1
tar xvf ../root.tar
```

で展開。 /home2/clients/excludes の中身は

```
home2
usr2
usr6
home
proc
```

このうち、 home2, proc は必須。 usr2, usr6 は研究室ユーザのホームがあるディレクトリ (もちろん NFS) で、これももちろん除外する必要がある。 tar にもうちょっと賢いオプションがあるかもしれないけど知らないので。

で、これを export する。 /etc/exports の中身は

```
/home2  *(rw,no_root_squash)
```

これは結構よろしくなくて、もうちょっと export する範囲とかちゃんと制限する必要がある。とりあえず、プライベートの中なのでこれですます。

で、 TL8 の場合には /etc/hosts.allow にもなんか書く必要があった。

で、カーネルソースもコピーされているので、

```
/home2/clients/test1/usr/src/linux
```

で

```
make menuconfig
make bzImage
```

する。menuconfig で、

```
kernel autoconfig
NFS root mount
NIC support
```

辺りを設定する必要あり。これくらいだったと思う。他のいろんなものは基本的にモジュールではなく、カーネル組み込みにしておく。モジュールにしておくとどこからいつロードするかが難しいからである。

新しいカーネルは、syslinux をおいたディレクトリに

```
cp /home2/clients/test1/usr/src/linux/arch/x86_64/boot/bzImage vmlinuz
```

でコピー。

```
cp -p /home2/clients/test1/usr/src/linux/.config napa-config-diskless
```

で、 config を保存しておく。napa はファイルサーバの名前。

さらに、 etc/fstab は変更する必要がある。

もとは

```
/dev/hda5               /                       ext2    defaults        1 1
/dev/hda1               /boot                   ext2    defaults        1 2
/dev/hda6               /home                   ext2    defaults        1 3
/dev/hdc1               /home2                  ext3    defaults        1 4
/dev/cdrom              /mnt/cdrom              iso9660 noauto,owner,ro 0 0
/dev/fd0                /mnt/floppy             auto    noauto,owner    0 0
none                    /proc                   proc    defaults        0 0
none                    /dev/pts                devpts  gid=5,mode=620  0 0
/dev/hda7               swap                    swap    defaults        0 0
#この後に NFS の記述
```

だったものを

```
192.168.1.11:/home2/clients/test   /            nfs     rw        0 0
none                    /proc                   proc    defaults        0 0
none                    /dev/pts                devpts  gid=5,mode=620  0 0
#この後に NFS の記述
```

にする。 root の指定は本当に必要なのかどうか今一つ疑問だけど。また、 eth0 は kernel autoconfig の設定をしたのでカーネルロード時に設定されるので、ifup-eth0 が実行される必要はない。するとなんか変になるかもしれないので、

etc/sysconfig/network で

```
mv ifcfg-eth0 original-ifcfg-eth0
```

を実行する必要あり。これくらいだったと思う。

4 syslinix.cfg の設定

root を nfs にするとかいろんな指定がいる。

```
DEFAULT vmlinuz
APPEND ip=dhcp root=/dev/nfs nfsroot=192.168.1.11:/home2/clients/test1 ip=192.168.1.160:192.168.1.
TIMEOUT 3
PROMPT 1
```

これくらいでいいはず。最初の ip=dhcp はいらないと思う。 192.168.1.11 はルートファイルをもってるファイルサーバの IP アドレス、 192.168.1.160 は自分 (ディスクレスノード) の IP アドレスである。

で、もう一度 USB メモリにこれを書き込んで、ディスクレスノードにいれてブートして、立ち上がればめでたしめでたし。

最初の NFS マウントが出来て、途中までいって read/write remount に失敗する場合は、 etc/mtab を消す必要があるかもしれない。

5 shutdown sequence の変更。

さて、無事に立ち上ががったとして、 shutdown/reboot できるかどうかが問題である。まあ、 diskless だからいきなり電源切ってもかまわないけど、reboot がコマンドでできないのはクラスタとしては不便である。しかし、shutdown は結構厄介である。つまり、あらゆるコマンドが NFS 先にしかないのに、通常の shutdown は途中で NFS unmount してネットワークを止めるようになっているからである。

面倒なので、 /etc/rc.d/rc[0,6].d が

```
K05atd
K05keytable
K14alsasound
K20nfs
K25sshd
K50xinetd
K60crond
K75netfs
K80random
K86nfslock
K90network
K98kparam
K99syslog
S00killall
S01halt
```

となっていたところを

```
K00cupsd
K05atd
K05keytable
K14alsasound
K20nfs
K25sshd
```

```
K50xinetd
K60crond
K65murasaki
K80random
K86nfslock
K88kparam
K89syslog
K91netfs
K92network
S00killall
S01halt
```

としてしまう。で、rc6.d のほうでは

```
K91netfs:
 umount /usr2
 umount /usr6
 rm /etc/mtab
 exit 0

K92network:
 rm -f /var/lock/subsys/network
 /sbin/reboot -i
 exit 0
```

netfs のほうはもうちょっと工夫がいる (root 以外の nfs マウントなものを外す、というようなスクリプトにするべき)。 network のほうは、まあ、とにかく reboot が通ればそれでいい。

rc0.d のほうは、少なくとも TL8 ではもうちょっと厄介であった。/sbin/poweroff が、実体は /sbin/reboot と同じくせに違う動作をするからである。これは現在のところ結局 K92network の中で止まるようにしてそこで電源を切ってしまっている。

6 マシン毎の設定について

6.1 USB メモリ

USB メモリのほうはこんなスクリプトを書いた (usbinit.csh)

```
#
# usbinit.csh
#
# create SYSLINUX USB stick with specified g6host id
#

/sbin/sfdisk --force /dev/sda << EOF
0,1014,6,*
EOF
/sbin/fdisk -l /dev/sda
/sbin/mkfs.msdos /dev/sda1
./syslinux /dev/sda1
```

```
ruby syslinuxedit.rb $1 > syslinux.tmp
cat syslinux.tmp
mcopy vmlinuz c:
mcopy syslinux.tmp c:syslinux.cfg
mcopy syslinux.tmp c:${1}
mdir c:
```

syslinuxedit.rb は単純な書き換えだけ

```
node=ARGV[0].to_i
ip = node + 128
print <<END
DEFAULT vmlinuz
APPEND ip=dhcp root=/dev/nfs nfsroot=192.168.1.11:/home2/clients/g6host#{node} ip=192.168.1.#{ip}
TIMEOUT 3
PROMPT 1
END
```

これで、

```
csh -f usbinit.csh 33
```

とかでブート用 USB メモリができる。

6.2 ルートファイルシステム

上で作った USB メモリは /home2/clients/g6hostxx というものがあることを仮定しているので、これを作る必要がある。テスト用のものから全部コピーしてもいいんだけど、共有できるものは共有したい。

が、 Linux は (少なくとも ext2 では) ディレクトリのハードリンクを禁止しているので、これが意外に面倒である。ソフトリンクではマウントしたディレクトリの外にいってしまうので、リンク先をアクセスできない。

しょうがないので、とりあえず /usr だけ共有にして、これは別に置いてマウントすることにした。 つまり、ディスクレスノードの /etc/fstab は例えばこんなの

```
192.168.1.11:/home2/clients/test   /            nfs     rw        0 0
192.168.1.11:/home2/clients/base.usr   /usr           nfs     rw        0 0
none                    /proc                   proc    defaults        0 0
none                    /dev/pts                devpts  gid=5,mode=620  0 0
```

にしておく。ここで root のマウントポイントは嘘だけど、これは使われないのか特に問題なさそうである。base.usr は usr の実体がある。で、各ディスクレスノード用のルートファイルシステムは、結局 /usr 以外全部を持つ。で、/usr はマウントポイント用のディレクトリだけ。

で、 その状態のルートファイルシステムを、 tar で g6host26.tar という名前でアーカイブしておく。これを使って新しいノード用のコピーを作るのが、以下の makensfroot.csh

```
# makensfroot.csh
#
# make copy of the basesystem
#directories to copy
```

```
set name = g6host${1}
mkdir $name
cd $name
echo extracting the tar file for all file systems
tar xf ../g6host26.tar
echo removing improper etc/mtab
rm etc/mtab
echo creating new
cat <<EOF > etc/sysconfig/network
NETWORKING=yes
PROFILENAME="No_Profile"
HOSTNAME=$name
DOMAINNAME=astron.s.u-tokyo.ac.jp
GATEWAY=192.168.1.96
GATEWAYDEV=eth0
FORWARD_IPV4=no
IPX=no
TIMESERVERATBOOT=no
EOF
echo making nfsroot for node $name end
```

この中で /etc/sysconfig/network を新しく作り、ここのホスト名を書いておく。これを、親ディレクトリで実行すれば新しいノード用のルートファイルシステムができる。 300MB くらいなので大して時間はかからない。

6.3 マザーボード BIOS の設定。

A8V の場合、 BIOS で

```
Boot -> Removable なんとかで USB Flush memory を優先にする
Boot -> device なんとか　でフロッピー (USB メモリになる) を優先にする
Boot -> config? で F1 を待たないようにする
```

でいいはず。ブート時間を短縮するには、使わないものを止める、特に

Advanced-¿ onboard で SATA とか RAID を止める

のが効果的。

7 問題点と解決方法

7.1 なにも考えないでリセットすると次のブートでこける

これは、単にマウントが /etc/mtab を見るのが悪いんだけど、、、/proc/mounts への soft link に変えるとか、そういう方法もあると思われる。