LPIC LEVEL 1 対策

～仮想化環境による実習環境整備補足資料～

Ver. 1.3

リナックスアカデミー矢越昭仁

2013/03/20

LPICレベル1技術解説無料セミナー」用補足資料。VirtualBoxに関する操作方法および補足事項

目次

はじめに

ここ数年、仮想化ソフトの技術革新は目覚ましく、PC のパフォーマンス向上も手伝い、ずいぶんと普及しています。自宅の PC 環境を壊すことなく手軽に Linux 環境を構築することが可能となりました。
この資料では、Oracle VirtualBox を、「LPIC 対策」に適用するために行うべき事をまとめています。

表記について

この資料では以下の表記としています。

・フォント

コンピュータの操作および設定ファイルはクーリエフォント(タイプライター風)を用います。

```
search t123006.la.net
nameserver 10.20.123.6
```

・プロンプト

コマンド入力例がある場合は、先頭はプロンプト($または#)で始めます。
$ は一般ユーザでの操作、#はルートユーザでの操作を表します。なおユーザ切り替え(su)は省略しています。

・強調(ボールド)

コマンド入力では、キーボードから入力する場合を、設定ファイルの場合は修正箇所など特に強調したい場合に**ボールド**を使います。

```
$ date
Mon Mar 5 12:32:41 JST 2012
```

・凡その作業時間

凡その作業時間とは、過去に同様の作業を経験した人が再度実行した場合にかかる時間を想定しています。つまり事前調査や試行錯誤の時間を含まない作業時間を指します。

Virtual Box のインストール

Virtual Box（以下 VBOX）サイトのダウンロードメニューから、必要なキットを入手します。例えば Windows で動作させるキットは「Virtual Box バージョン for Windows hosts」となり、x86 が 32bit 版、amd64 が 64bit 版となります。

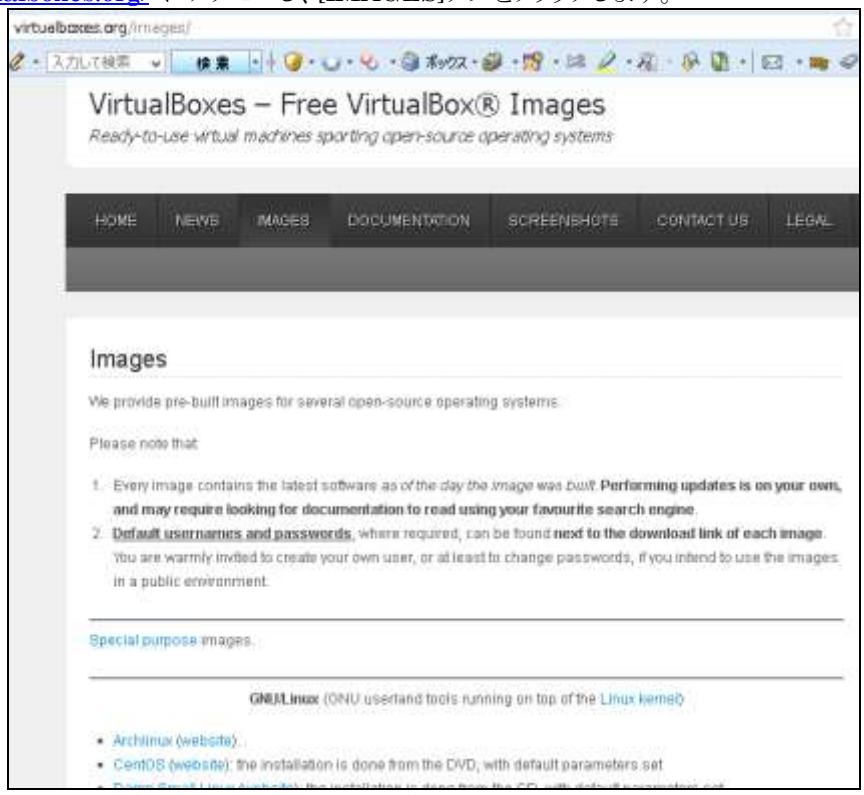
https://www.virtualbox.org/

VBOX のインストールは簡単ですが、途中で仮想的なハードウエアデバイスを追加するため、システム管理者としての許可を求められます。

ゲスト OS ファイルのダウンロード

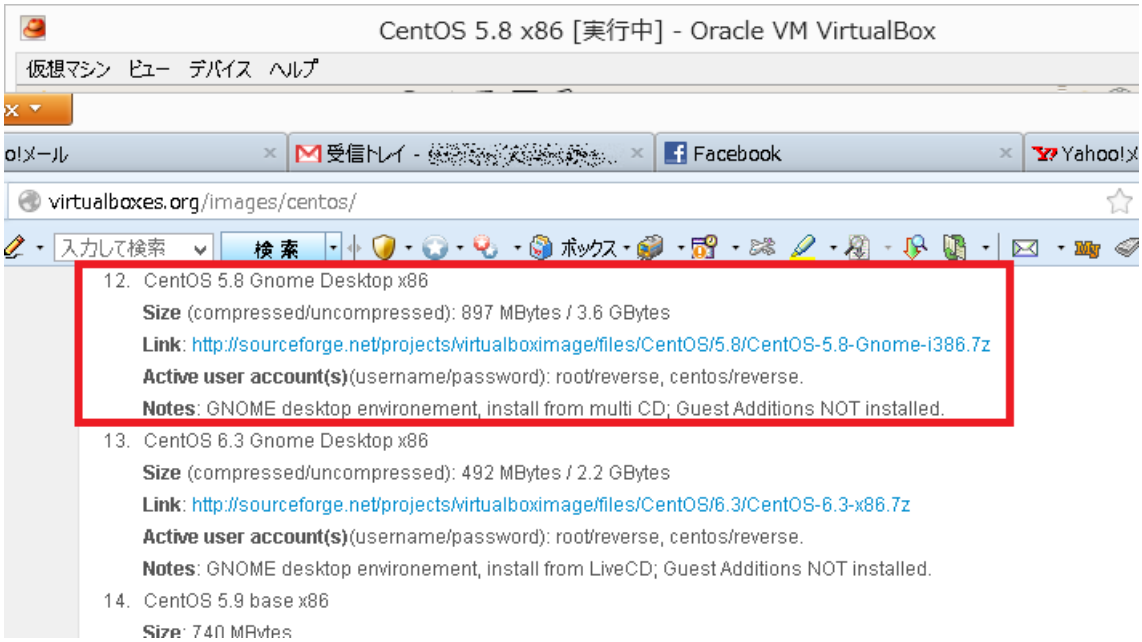
VBOX の普及を促進するため、様々な Linux ゲスト OS がネットワーク上で公開されています。これらのファイルはダウンロードするだけで利用でき、とても手軽です。

http://virtualboxes.org/ にアクセスし、[IMAGES]タブをクリックします。



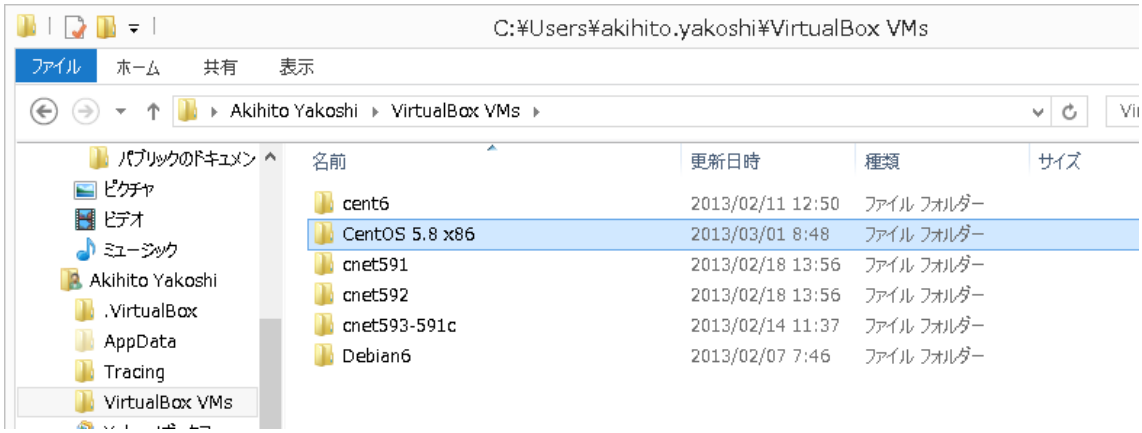
ディストリビューションごとにリンクがあります、この資料では CentOS を選択します。

ディストリビューションごとのリンクでは、さらにバージョンごとに数多くのファイルが格納されています。ファイルの形式がまちまちなので、インストール可能な形式を選びます。VBOX を起動後に OS のバージョンアップも可能なので、多少古いバージョンでも構いません。ただし CentOS 5.x と 6.x 間は大きく異なるためバージョンアップによる移行はできません。



この例では、CentOS の 5.8 (32bit 版)を選択しています。リンクをクリックするとファイルのダウンロードが始まります。
このリンクの下にログインに必要な情報 Active user account が記述されているので、忘れずにメモを取っておきます。この例ではユーザ名が root パスワードが reverse となっています。

ダウンロードが完了したら、ファイルを展開します。展開する先はドキュメント下の VirtualBox VMS フォルダ下にしておきます。圧縮したファイルに比べ、展開後は数 GB と非常に大きくなるため、十分な HDD 空き容量が必要です。



展開したフォルダの .vbox ファイルをダブルクリックすると仮想環境が立ち上がります。VBOX のバージョン、PC の型式によってはエラーが発生する場合があります。

この例では、USB コントローラの不整合が発生しています。仮想環境での設定に関するだけで、PC の USB に障害が発生している訳ではありません。

この場合は、一旦 VirtualBox マネージャを起動し当該仮想マシンを選択し、設定ボタンをクリックします。
設定ボタンはマネージャ上側の歯車アイコンです。

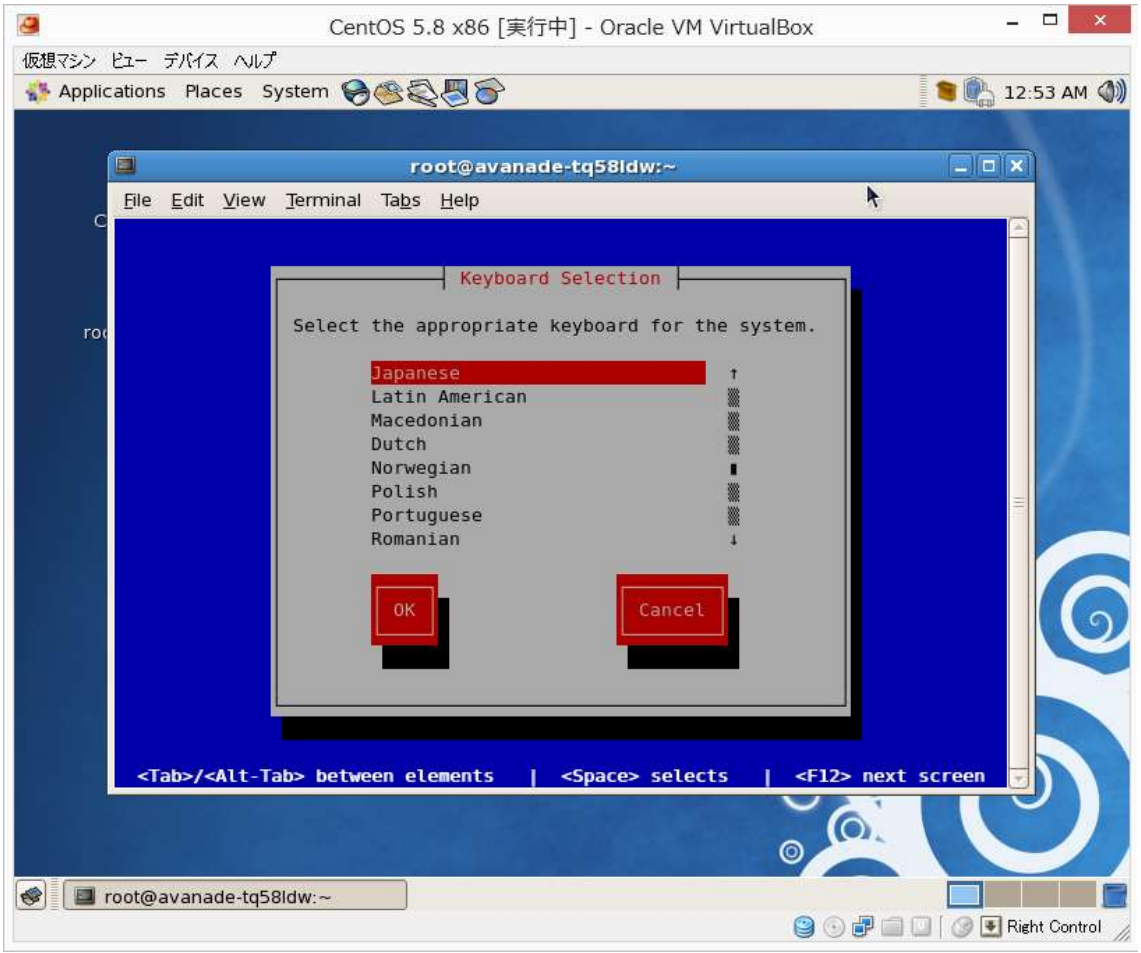
設定項目の一覧から、USB をクリックし「□USB コントローラを有効化(U)」のチェックを外し保存します。

再び仮想マシンをダブルクリックすれば仮想マシンが起動します。先にメモしたユーザ名とパスワードでログインできます。

多くの仮想マシンは英語環境となっており、キーボードの割当が異なる場合があります。CentOS では起動直後に仮想端末()から、

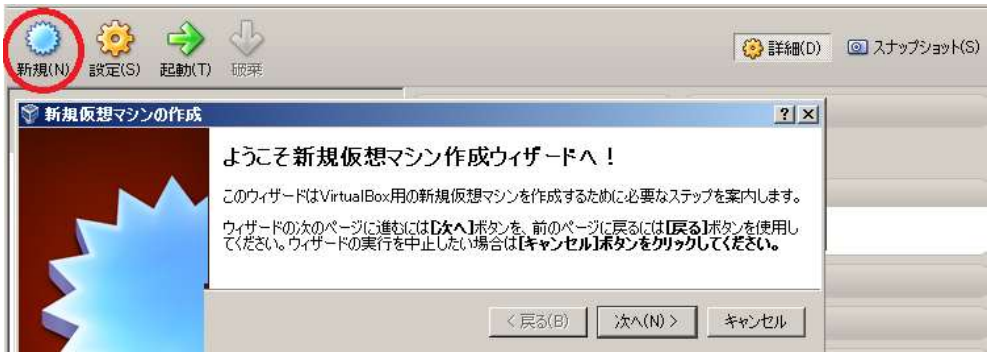
```
# setup
```

コマンドを使ってキーボードの割当を変更します。



ゲスト OS の導入

ここからは、OS のインストールから行う場合の手順を解説します。ゲスト OS の入手については、他に多くの資料があるため、そちらを参照してください。ここでは VBOX でゲスト OS を導入する方法を解説します。ゲスト OS の ISO ファイルを入手したら、VBOX を起動し、「新規」ボタンをクリック、新規仮想マシン作成ウィザードを起動します。



続いて、「名前(A)」に仮想マシンの名称、簡易インストールの場合は「OS タイプ」を指定します。簡易インストールはユーザに代わって、VBOX がゲスト OS のインストーラからの質問に回答します。
その為指定したバージョンと、実際に導入する OS が合致している必要があります。異なる場合は、一般名称(Linux 2.6 など)を指定します。

左図の例では、仮想マシン名に Fedora150、OS 名は Linux とし、バージョンにはディストリビューション Fedora(32bit)を指定しています。

メモリは推奨値でかまいません。インストール後に調整が可能です。

ハードディスク(実際には VBOX のファイル)を指定します。
OS を導入する際には。起動ディスクを指定します。

仮想ファイル形式は、VMware や Parallels など他社のフォートも利用可能ですが、既存のデータを参照する以外は VDI がよいでしょう。

割り当て方法は Dynamic（動的：仮置きでファイルを作成し、実際にアクセスがあった場合にディスクを消費）と、Static（静的：この時点で、ディスクにダミーデータを使いディスク容量を確保）があります。
パフォーマンスは Static が良いですが実際に多くのディスクを消費します。

割り当てる場所と、大きさを指定します。
場所はゲスト OS に係る定義内容の格納先が既定となります。
サイズは 8.0GB が標準ですが、Dynamic 割り当ての場合は、実際に消費した容量しか使いません。

仮想ディスクに関する概要が表示されます。
問題なければ[Create]で先に進みます。

ゲスト OS 全体に関する確認画面が表示され、[Create]で実際に仮想マシンが作成されます。

以上で、仮想マシンが用意され VBOX マネージャに一覧表示されます。起動は仮想マシンを選び、起動（⇒アイコン）をクリックします。

起動した最初の1回だけ、インストールウィザードが表示されます。

インストールメディアを指定しゲスト OS のインストールが開始されます。

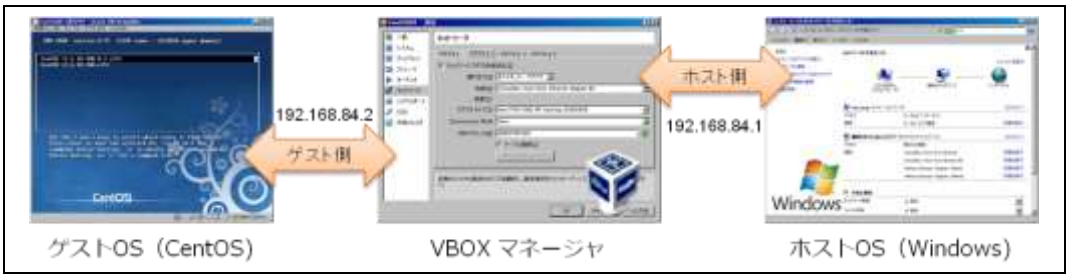
OS のインストールは各 OS のサイトや資料を参照してください。

VBOX 操作

VBOX での基本的な操作について解説します。

ネットワークの調整

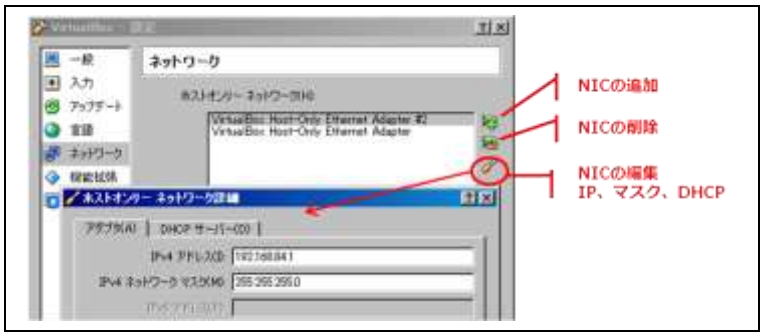
VBOX では、仮想マシンに NIC を追加する毎にホスト OS 上に仮想的な NIC を導入します。導入後に NIC の設定を修正した場合などは、それら相互に矛盾が生じ動作不良となる場合があります。



矛盾が生じた場合は、以下の順に確認・設定します。

1. ホスト OS
   ホスト OS の設定を基準とします。IP アドレスと MAC(物理)アドレス両方を確認してください。プロパティから詳細を選択するか、コマンドプロンプトより ipconfig/all で確認する事が出来ます。

2. VBOX マネージャ
   ネットワーク接続の新規作成（IP アドレスの割り当て）はメニュー：ファイル＞環境設定（P）から行います。



   MAC アドレスの設定は、仮想マシンが停止している状態で設定（S：歯車アイコン）をクリックして行います。

   ホスト OS = VBOX マネージャである事を確認してください。

3. ゲスト OS
   上記の設定を元に、ゲスト OS の設定を行います。CentOS の場合は、/etc/sysconfig/network-scripts/ifcfg-ethX(X は VBOX マネージャで指定したアダプタ番号)で設定します。
   VBOX マネージャのアダプタと同じネットワークアドレスで、あることを確認します。
   ゲスト OS≠VBOX マネージャであることを確認します。

例）　ホスト/VBOX　　192.168.84.1/24
　　ゲスト OS(CentOS) 192.168.84.2/24

共有フォルダ

VBOX ではゲスト OS とホスト OS の間でデータを共有できる「共有フォルダ(以下、vboxsf)」を簡単に作る事ができます。

VBOX マネージャ側

設定(S:歯車アイコン)から、共有フォルダを呼び出します。

設定内容:

- フォルダのパス
  ホスト OS(Windows)側のフォルダ名を指定します。
- フォルダ名
  ゲスト OS から参照される vboxsf ファイルシステムの論理名を指定します。
- 自動マウント
  ゲスト OS 起動時にマウントします。
- 永続化する
  次回起動時にも利用できるよう、上記内容を保存します。

ゲスト OS 側

初めて vboxsf を利用する場合は、ゲスト OS にドライバを組み込む必要があります。
VBOX に添付されている、専用ドライバをコンパイルし、ゲストOSに組込む必要があり、前提パケージとして以下のものが予めインストールされている必要があります。

・X Window System
・kernel-devel、kernel-headers
・GCC (コンパイラ)

続いての手順は、仮想マシンのメニューからドライバ＞Guest Additions のインストールを選択します。

この時点で、VBOX に内包されている ISO ファイルが仮想マシン上の CD として有効になります。
CD をマウントし、当該ツール(VBoxLinuxAdditions.run)を実行し、再起動します。

```
[root@localhost ~]# mount -r /dev/cdrom /media
[root@localhost ~]# /media/VBoxLinuxAdditions.run
Verifying archive integrity... All good.
Uncompressing VirtualBox 4.1.2 Guest Additions for Linux.........
VirtualBox Guest Additions installer
Removing existing VirtualBox DKMS kernel modules     [  OK  ]
Removing existing VirtualBox non-DKMS kernel modules [  OK  ]
```

```
Building the VirtualBox Guest Additions kernel modules
Not building the VirtualBox advanced graphics driver as this Linux version is
too old to use it.
Building the main Guest Additions module             [  OK  ]
Building the shared folder support module            [  OK  ]
Doing non-kernel setup of the Guest Additions        [  OK  ]
Starting the VirtualBox Guest Additions              [  OK  ]
Installing the Window System drivers                 [FAILED]
(Could not find the X.Org or XFree86 Window System.)
[root@localhost ~]# umount /media
[root@localhost ~]# shutdown -r now
```

X を導入していない環境の場合、最後の GUI ツール導入部分で FAILED になりますが、動作に支障はありません。
再起動後、自動マウントされている場合は、先の共有フォルダが /media/sf_(フォルダ名)でマウントされます。

```
[root@localhost ~]# df
Filesystem        1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                    7491040   2287316   4817064  33% /
/dev/sda1             101086     18785     77082  20% /boot
tmpfs                 127492         0    127492   0% /dev/shm
share              156287996 150204560   6083436  97% /media/sf_share

[root@localhost ~]# mount
/dev/mapper/VolGroup00-LogVol00 on / type ext3 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/sda1 on /boot type ext3 (rw)
tmpfs on /dev/shm type tmpfs (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
share on /media/sf_share type vboxsf (gid=501,rw)
```

vboxsf はファイルシステムとして拡張されているので、 mount コマンドは以下のようになります。

```
# mount -t vboxsf share /media/sf_share
```

仮想ディスクの追加

仮想ディスクは IDE(/dev/hd?)と SATA(/dev/sd?)の 2 系統があり、追加・削除は仮想マシンが停止している状態で行います。(コントローラ上の、+アイコンが追加。CD と HDD の 2 種)

プロパティにより接続場所を確認することができます。ゲスト OS 側では、/dev/disk/by-path からリンクをたどることで、デバイス名を同定する事が可能です。

```
[root@localhost by-path]# pwd
/dev/disk/by-path
[root@localhost by-path]# /bin/ls -g
total 0
```

```
lrwxrwxrwx 1 root         9 Jun 18 03:13 pci-0000:00:01.1-ide-0:0 -> ../../hdc
lrwxrwxrwx 1 root         9 Jun 18 03:13 pci-0000:00:0d.0-scsi-0:0:0:0 -> ../../sda
lrwxrwxrwx 1 root        10 Jun 18 03:13 pci-0000:00:0d.0-scsi-0:0:0:0-part1 -> ../../sda1
lrwxrwxrwx 1 root        10 Jun 18 03:13 pci-0000:00:0d.0-scsi-0:0:0:0-part2 -> ../../sda2
lrwxrwxrwx 1 root         9 Jun 18 03:13 pci-0000:00:0d.0-scsi-1:0:0:0 -> ../../sdb
```

仮想マシンのコピー

仮想マシンをコピーするには 2 つの方法があります。1つはクローンで、これはハードウエア固有情報を修正することで別のマシンとして利用する事ができます。もうひとつはスナップショットで、いわゆるシステムのバックアップに相当します。これらの操作はゲストOSが停止している状態で行う必要があります。

VMクローン

指定したVMに関するファイルを全て別名でコピーし、システム固有情報に関する部分は変更を行います。なお大量のファイルをコピーする為、時間がかかります。

メニューの「仮想マシン(M) > Clone...」を選びます。

VM名を指定します、省略値は「コピー元VM名 clone」となります。

クローン方法を指定します、Full Clone は全てのファイルをコピーします。
Linked Clone は共有可能なファイルは共有することでファイルの容量と作成時間を節約します。

クローンが完了すると、一覧にVMが追加されます。

スナップショット

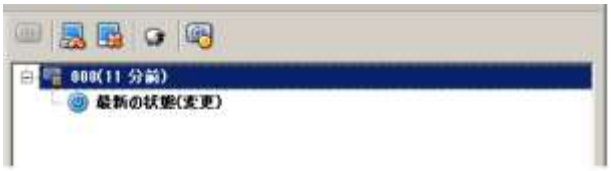
スナップショットを取得しておけば、システム設定に失敗した場合でも、取得した時点まで復帰する事ができます。

一覧上のスナップショットをクリックし、スナップショットの状況を表示させます。

現在の状態を選択し、カメラアイコンをクリックします。

スナップショット(カメラに停止マーク■)アイコンが作成されます。

この状態で起動した後は、スナップショットまで立ち戻る事ができます。



システム停止後、スナップショットを選択し、復元アイコン(カメラに^)で、それまでのシステム変更を無効としスナップショットを取得した時点に戻します。
不要なスナップショットは削除アイコン(カメラに×)で削除できます。

実習例

VBOX を使った実習例

パーティションの追加

ディスクの追加は「仮想ディスクの追加」にて行い、対応するデバイスファイルについてパーティションの操作を行います。IDE のディスクであれば、hda, hdb, hdc の順に、SATA(SCSI)ディスクであれば sda, sdb, sdc の順にデバイス番号と対応します。
以下の例は、予め作成した約 600MB の仮想 sdb ディスクを、3つのパーティションに分割した場合です。

```
[root@localhost ~]# fdisk /dev/sdb
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course, the previous
content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-76, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-76, default 76): +200M

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (26-76, default 26):
Using default value 26
Last cylinder or +size or +sizeM or +sizeK (26-76, default 76): +200M

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 3
First cylinder (51-76, default 51):
Using default value 51
Last cylinder or +size or +sizeM or +sizeK (51-76, default 76): +200M

Command (m for help): p

Disk /dev/sdb: 629 MB, 629145600 bytes
255 heads, 63 sectors/track, 76 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot    Start         End      Blocks   Id   System
/dev/sdb1             1          25      200781   83   Linux
/dev/sdb2            26          50      200812+  83   Linux
/dev/sdb3            51          75      200812+  83   Linux

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

ソフトウエア RAID の実装

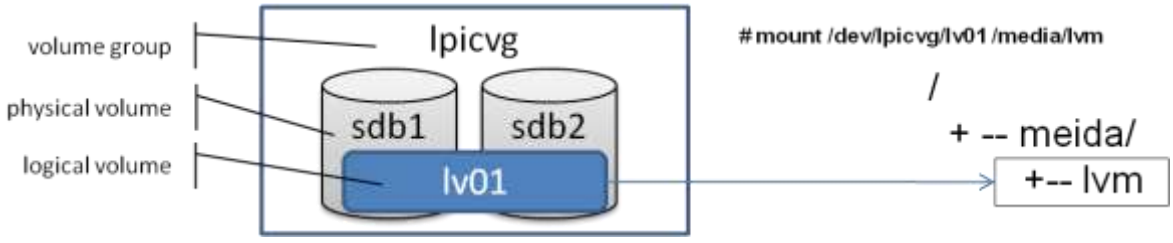
mdadm(8)を使ったフトウエア RAID の構築および、メンテナンスの実習事例。
(予め仮想ディスク sdb1,2,3 を用意。「パーティションの追加」参照)

```
[root@localhost ~]# mdadm  --create /dev/md0 --level 1 --raid-devices 2
--spare-devices 1 /dev/sdb[123]
[root@localhost ~]# cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sdb3[2](S) sdb2[1] sdb1[0]
      200704 blocks [2/2] [UU]

unused devices: <none>

# 作成した RAID デバイスを Ext3 で初期化 (ルートの取り分は 0%)
[root@localhost ~]# mkfs -t ext3 -m 0 /dev/md0
mke2fs 1.39 (29-May-2006)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
50200 inodes, 200704 blocks
0 blocks (0.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=67371008
25 block groups
8192 blocks per group, 8192 fragments per group
2008 inodes per group
Superblock backups stored on blocks:
       8193, 24577, 40961, 57345, 73729

Writing inode tables: done5
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 33 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.

[root@localhost ~]# mdadm --detail /dev/md0
/dev/md0:
       Version : 0.90
  Creation Time : Mon Jun 18 12:23:17 2012
     Raid Level : raid1
     Array Size : 200704 (196.03 MiB 205.52 MB)
  Used Dev Size : 200704 (196.03 MiB 205.52 MB)
   Raid Devices : 2
  Total Devices : 3
Preferred Minor : 0
    Persistence : Superblock is persistent

    Update Time : Mon Jun 18 12:26:08 2012
          State : clean
 Active Devices : 2
Working Devices : 2
 Failed Devices : 1
  Spare Devices : 0

           UUID : 7f31fa51:4750d719:47356e35:e397adb2
         Events : 0.10

    Number   Major   Minor   RaidDevice State
       0       8       17        0      active sync   /dev/sdb1
       1       8       19        1      active sync   /dev/sdb3

       2       8       18        -      faulty spare   /dev/sdb2
```

LVM

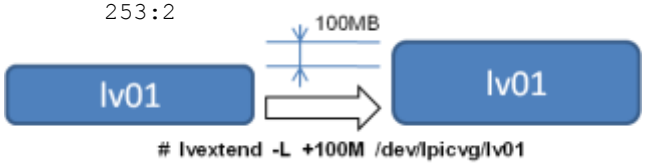
LVM の新規作成

```
[root@localhost ~]# pvcreate /dev/sdb1
WARNING: software RAID md superblock detected on /dev/sdb1. Wipe it? [y/n] y
  Wiping software RAID md superblock on /dev/sdb1.
  Writing physical volume data to disk "/dev/sdb1"
  Physical volume "/dev/sdb1" successfully created
[root@localhost ~]# pvcreate -f /dev/sdb2
  Wiping software RAID md superblock on /dev/sdb2.
  Writing physical volume data to disk "/dev/sdb2"
  Physical volume "/dev/sdb2" successfully created

[root@localhost ~]# vgcreate lpicvg /dev/sdb[12]
  Volume group "lpicvg" successfully created
[root@localhost ~]# pvs
  PV       VG          Fmt  Attr PSize   PFree
  /dev/sda2  VolGroup00 lvm2 a--    7.88G     0
  /dev/sdb1  lpicvg     lvm2 a--  192.00M 192.00M
  /dev/sdb2  lpicvg     lvm2 a--  192.00M 192.00M
[root@localhost ~]# vgs
  VG       #PV #LV #SN Attr   VSize   VFree
  VolGroup00   1   2   0 wz--n-  7.88G     0
  lpicvg       2   0   0 wz--n- 384.00M 384.00M

[root@localhost ~]# lvcreate -L 240 -n lv01 lpicvg
  Logical volume "lv01" created
[root@localhost ~]# mkfs -t ext2 -j /dev/lpicvg/lv01
mke2fs 1.39 (29-May-2006)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
61440 inodes, 245760 blocks
12288 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=67371008
30 block groups
8192 blocks per group, 8192 fragments per group
2048 inodes per group
Superblock backups stored on blocks:
      8193, 24577, 40961, 57345, 73729, 204801, 221185

Writing inode tables: done0
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 24 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
[root@localhost ~]# mount /dev/lpicvg/lv01 /media/lvm
[root@localhost ~]# df /media/lvm
Filesystem        1K-blocks      Used Available Use% Mounted on
/dev/mapper/lpicvg-lv01
                     238003     6176    219539   3% /media/lvm
```



LVM の拡張

```
[root@localhost ~]# vgextend lpicvg /dev/sdb3
  Volume group "lpicvg" successfully extended
[root@localhost ~]# lvextend -L +100M /dev/lpicvg/lv01
  Extending logical volume lv01 to 340.00 MB
  Logical volume lv01 successfully resized
[root@localhost ~]# umount /media/lvmcvg
[root@localhost ~]# e2fsck -f /dev/lpicvg/lv01
e2fsck 1.39 (29-May-2006)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/lpicvg/lv01: 11/61440 files (9.1% non-contiguous), 13933/245760 blocks
[root@localhost ~]# resize2fs /dev/lpicvg/lv01
resize2fs 1.39 (29-May-2006)
Resizing the filesystem on /dev/lpicvg/lv01 to 348160 (1k) blocks.
The filesystem on /dev/lpicvg/lv01 is now 348160 blocks long.

[root@localhost ~]# mount /dev/lpicvgK/lv01 /media/lvm
[root@localhost ~]# df /media/lvm
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/mapper/lpicvg-lv01
                      337041      6168    313465   2% /media/lvm
[root@localhost ~]# lvdisplay lpicvg
  --- Logical volume ---
  LV Name                /dev/lpicvg/lv01
  VG Name                lpicvg
  LV UUID                q7yBJt-ruIy-EIzE-DvVg-sSg2-ORgF-I7SgNa
  LV Write Access        read/write
  LV Status              available
  # open                 1
  LV Size                340.00 MB
  Current LE             85
  Segments               2
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:2
```



lvextend -L +100M /dev/lpicvg/lv01