

RHEL3/4 之 Logical Volume Manager

大綱：

LVM 概念

LVM 運作機制及重要名詞

實例演練 1 利用 Disk Druid 實作 LVM

LVM 指令

實例演練 2 建立 LVM 的檔案系統 (建立/lvmdata 檔案系統，其大小為 30MB)

實例演練 3 動態放大檔案系統 (將/lvmdata 檔案系統由 30 MB 放大為 40MB)

實例演練 4 動態加大虛擬 HD (Volume Group) 大小

實例演練 5 刪除 VG

從 RedHat Linux 8.0 開始便可以使用 Logical Volume Manager (LVM, 邏輯磁區管理員) 來做硬碟空間的分配。什麼是 LVM 呢? LVM (Logical Volume Manager) 原先運用於 IBM AIX Unix 系統, 是一種分配硬碟空間到邏輯磁區的方法, 這種方式比傳統分割區分配更容易做容量的調整, 它讓眾多硬碟機結合、彷彿融為一體, 可動態放大檔案系統的機制, 是讓 Linux 邁向企業高階伺服器的一大利器。

LVM 概念

使用 LVM, 硬碟中的分割區必須加入『Logical Volume Group 邏輯磁區群組 (簡稱 VG)』, 這種分割區在 LVM 中稱為『Physical Volume 實體磁區 (簡稱 PV)』, 各位可以把 VG 想成是一個大硬碟, 然後再從其中切出一塊空間『Logical Volume 邏輯磁區 (簡稱 LV)』對應至系統的檔案系統。

除了 /boot 分割區, 其餘的檔案系統皆可在 Volume Group 在這顆虛擬的大硬碟中。/boot 分割區不可以在 VG 中, 因為開機管理程式將無法讀取到它。假如您想要 / 檔案系統在 VG 中, 您必須另外建立一個 /boot 分割區, 使其不屬於 VG 的一員(圖 1)。

VG 可以分割為幾個『Logical Volume 邏輯磁區』, 將會指定它們的掛載點, 例如 /home 與 /, 以及檔案系統類型, 如 ext3。當某個檔案系統的容量全滿時, VG 中的剩餘空間可以增加到此檔案系統所對映的邏輯磁區以增大此檔案系統大小。

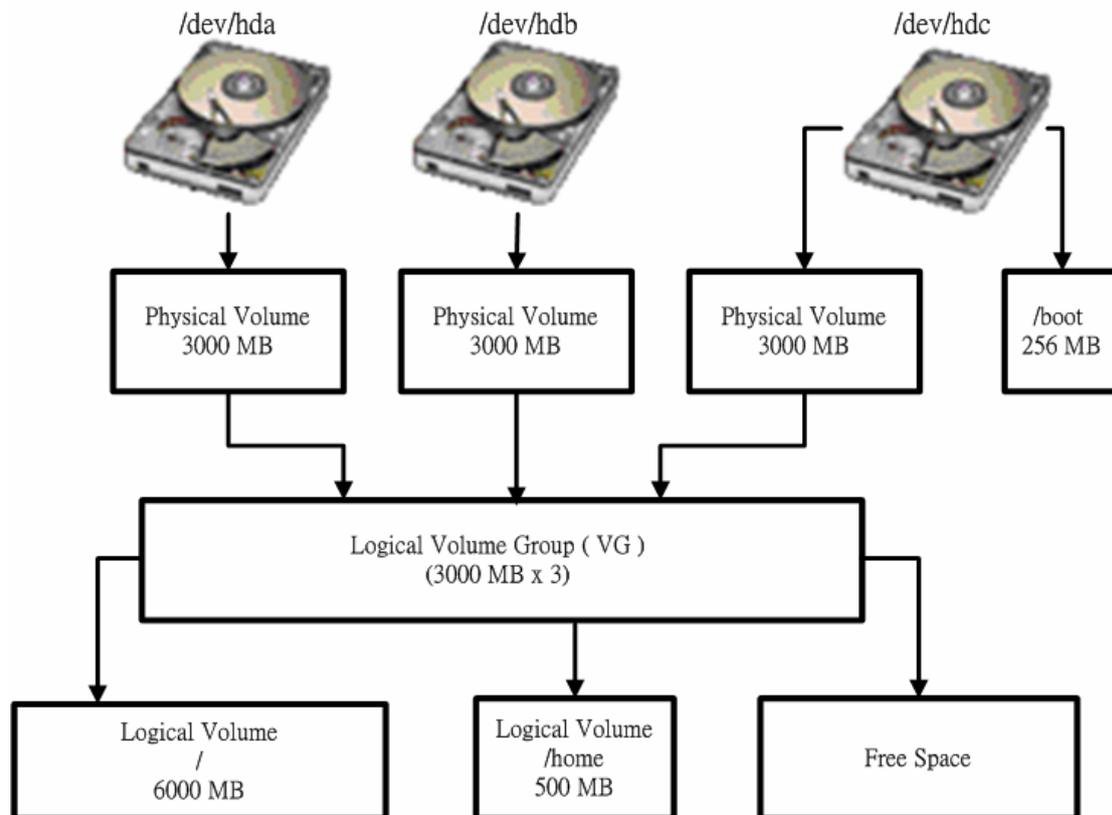


圖 1：Linux LVM 與檔案系統之間的關係圖

LVM 運作機制及重要名詞

接下來我們進一步介紹建立 LVM 的機制。首先，我們得先認識幾個名詞。LVM 機制原先為 IBM AIX 特有的硬碟管理機制，它的最主要的目的是為了克服 Unix 作業系統規劃硬碟空間給某個檔案系統之後，就無法再改變此檔案的大小。例如：各位可能當初在安裝系統時，可能認為/home 只要 500MB 就足夠，後來因為使用者愈來愈多，原先規劃的空間可能就不敷使用。

傳統的解法就是將原 /home 的資料備份出來，重新切割一塊新的分割區，格式化成新的檔案系統，再將原有的資料還原回去。即使現有的硬碟還有剩餘的空間，你也無法馬上利用這些空間，還是得重新切個新的分割區。但 LVM 利用在實體的硬碟和檔案系統加一層「**Logical 的對映機制**」來達到動態放大檔案系統的功能，進而解決傳統 Unix 所面臨的問題。

LVM 的重要名詞 (VG、PV、LV)：

筆者先介紹 VG 及 PV 這兩個名詞。在原先 IBM AIX 的紅皮書中的定義中，VG 是 LVM 機制最重要的東西，就是一堆 PV 的組合，一個 VG 可能包含一個 PV 或數個 PV。這樣解釋，各位一定聽不懂，筆者當年接觸 AIX 時也是搞得一個頭兩個大，覺得有點像用另一個不懂的名詞 (PV) 來解釋不懂的名詞 (VG)。

換個角度來看，其實各位可以把「VG 想成原先 Linux 實體觀念中的一顆硬碟」。而在實體世界中硬碟是由一個或多個分割區組成，在 LVM 中 VG 是由一個或一個以上的 PV 組成。所以在 Linux LVM 機制中，「PV 其實是由硬碟中的分割區所轉換變成的」。

那什麼是 LV，可不是那個服飾名牌 LV，當然搞懂了 LV 也不見得會幫你/妳賺到 LV 的包包，不過倒是可以幫你的 Linux 檔案系統 加上一件華麗的衣裳。好了言歸正傳，「LV 是 VG 中的一塊空間，就像原來 Linux 中的分割區 /dev/hda#」，是用來對應到 (mount) 系統中的某個目錄 (mount-point)。

- **Physical Volume (PV)**: 在 AIX 原有的定義將實體的硬碟稱為 PV，在 Linux 上各位可想成原先的一個分割區就是一個 PV。
- **Volume Group (VG)**: 就是一堆 PV 的組合，也就是說一個 VG 可能包含一個 PV 或數個 PV。
- **Physical Extent (PE)**: 當數個 PV 組合成一個 VG 時，LVM 會在所有的 PV 做類似格式化的工作，將每個 PV 切成一塊一塊的空間，這一塊一塊的空間就稱為 PE，通常是 4MB。
- **Logical Volume (LV)**: 由一群 LE 組成，在 LVM 的機制下一個檔案系統 (假設是 /home filesystem) 實際對應的不是 hda#，而是一個 LV。
- **Logical extent (LE)**: LV 的組成單位，它的大小為 PE 的倍數，通常為 1 : 1 的關係。

筆者將 LVM 中最重要的三個名詞及其意義整理簡列如下表：

表 1：PV、VG、LV 比較表

名詞	意義	作業系統相對應裝置
PV	實體分割區(分割區)	/dev/hda#
VG	虛擬硬碟(磁區群組)	/dev/<VG name>/ 目錄
LV	虛擬分割區(邏輯群組)	/dev/<VG name>/<LV name>

下圖(圖 2)是筆者用 Linux 的角度來看 LVM，所畫的 Linux LVM 的架構圖。

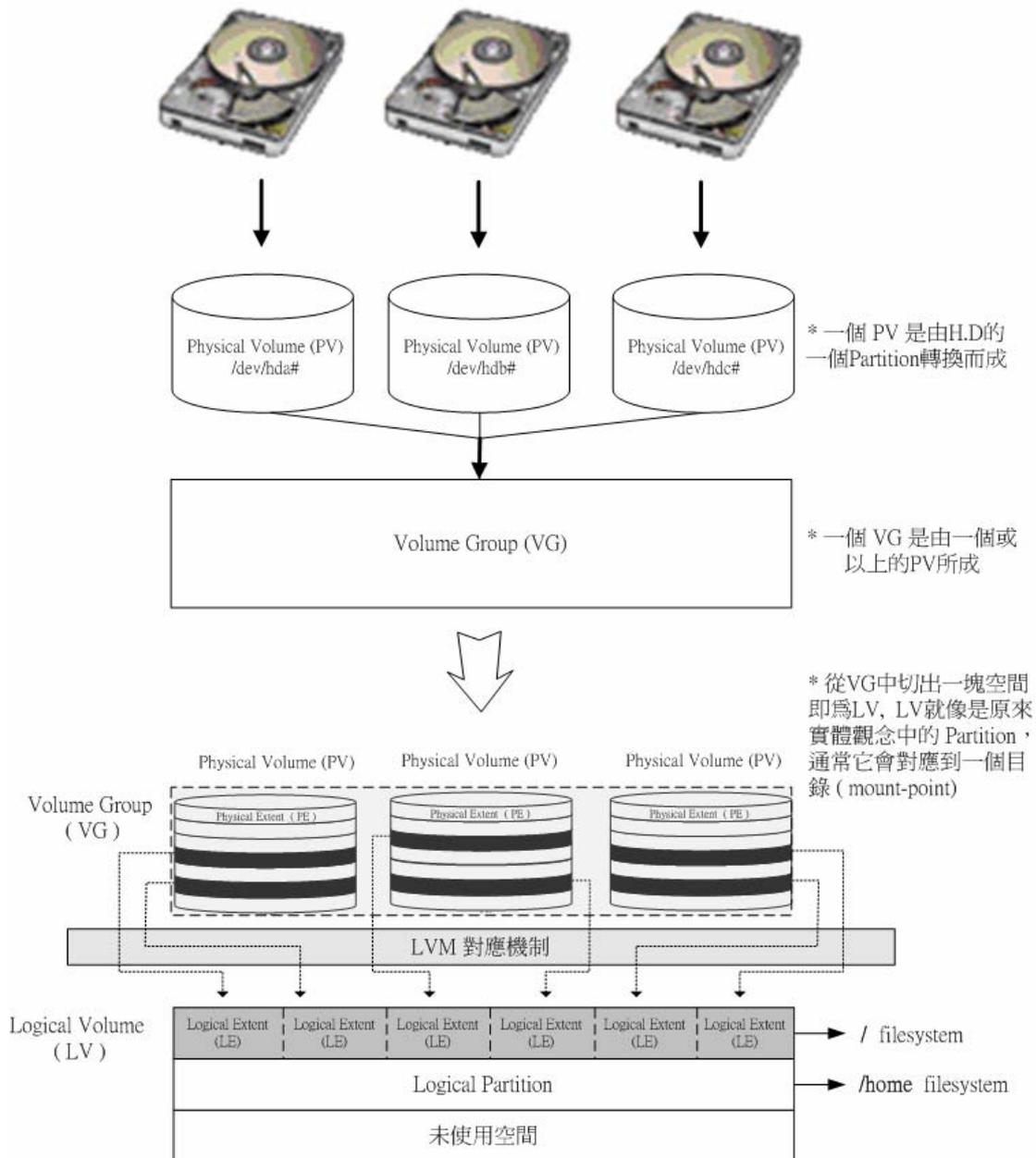


圖 2 : Linux LVM 架構圖

實例演練 1 利用 Disk Druid 實作 LVM

有兩種方式可以建立 LVM 的機制，一是藉由 Disk Druid (只有安裝時才提供)，另法是利用 LVM 指令來建立管理。下面為各位示範是利用 **Disk Druid** 來實作 LVM，最後示範如何動態放大 `/home` 檔案系統。

目的：利用 Disk Druid 實作 LVM，並學習使用如何動態放大 `/home` 檔案系統

測試環境：

RHEL 3/4 (亦適用於 Fedora Core 1/2/3/4)

分割區規劃：

- /boot 256 MB (/dev/hda1 傳統 Linux 檔案系統)
- 3 個 3GB 的 PV (/dev/hda2、/dev/hda3、/dev/hda4) 組合成一個虛擬硬碟 rootvg
- rootlv =6000MB → / 檔案系統
- homelv=500MB → /home 檔案系統
- swaplv=500MB → SWAP 檔案系統

步驟：

1./boot 256 MB → Linux 分割區 (圖 3、4)

(/boot 分割區不可以在 LVM 中，因為開機管理程式將無法讀取到它)



圖 3：利用 Disk Druid 來分割磁碟 (Disk Druid 只有在安裝時才可使用)

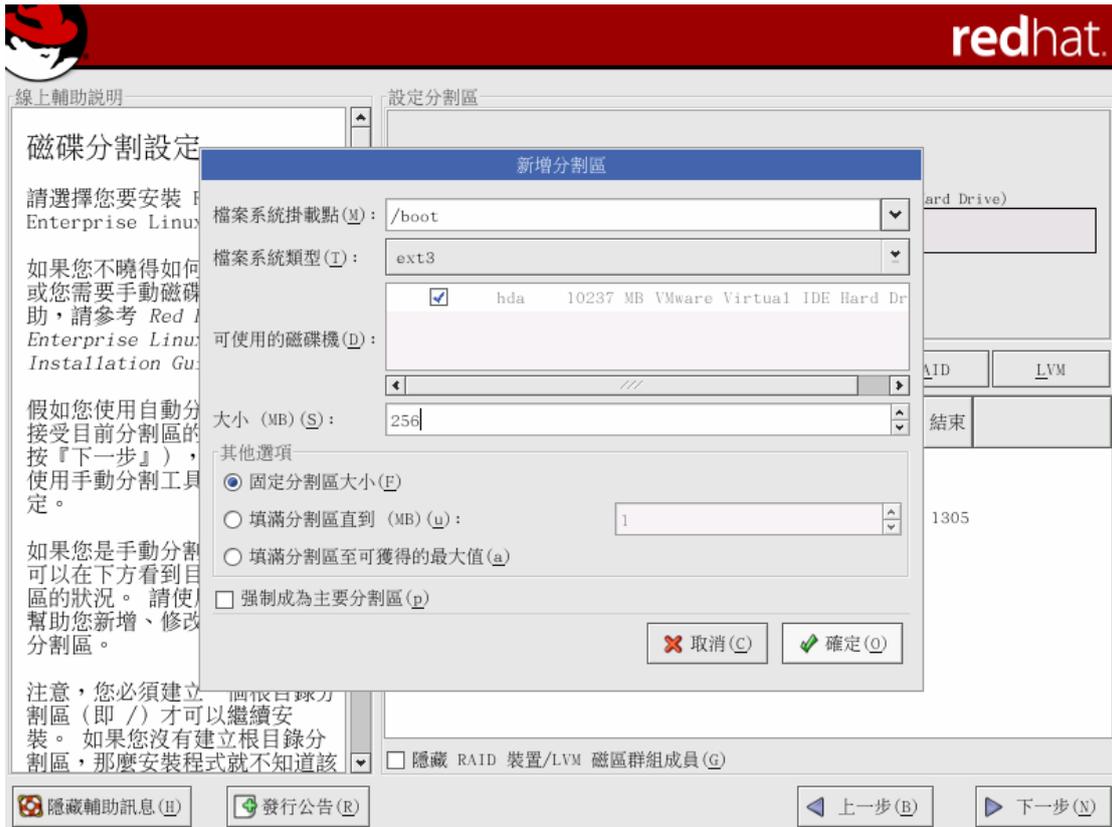


圖 4：新增 /boot 分割區

2. 然後切出三個大小 3000MB 分割區，hda2、hda3、hda4，檔案系統類型為 LVM，此步驟相當於建立三個 Physical Volume，最後這個硬碟還剩下 988 MB（圖 5、6）。

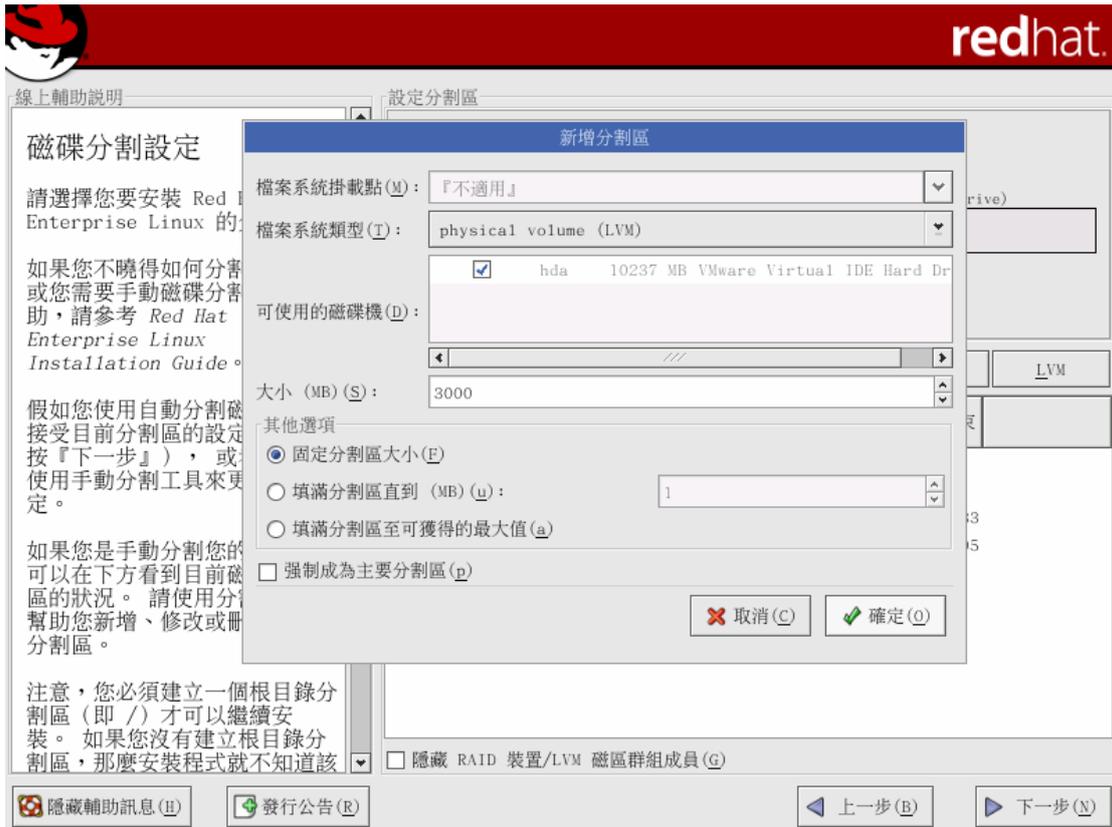


圖 5：新增 Physical Volume (PV)

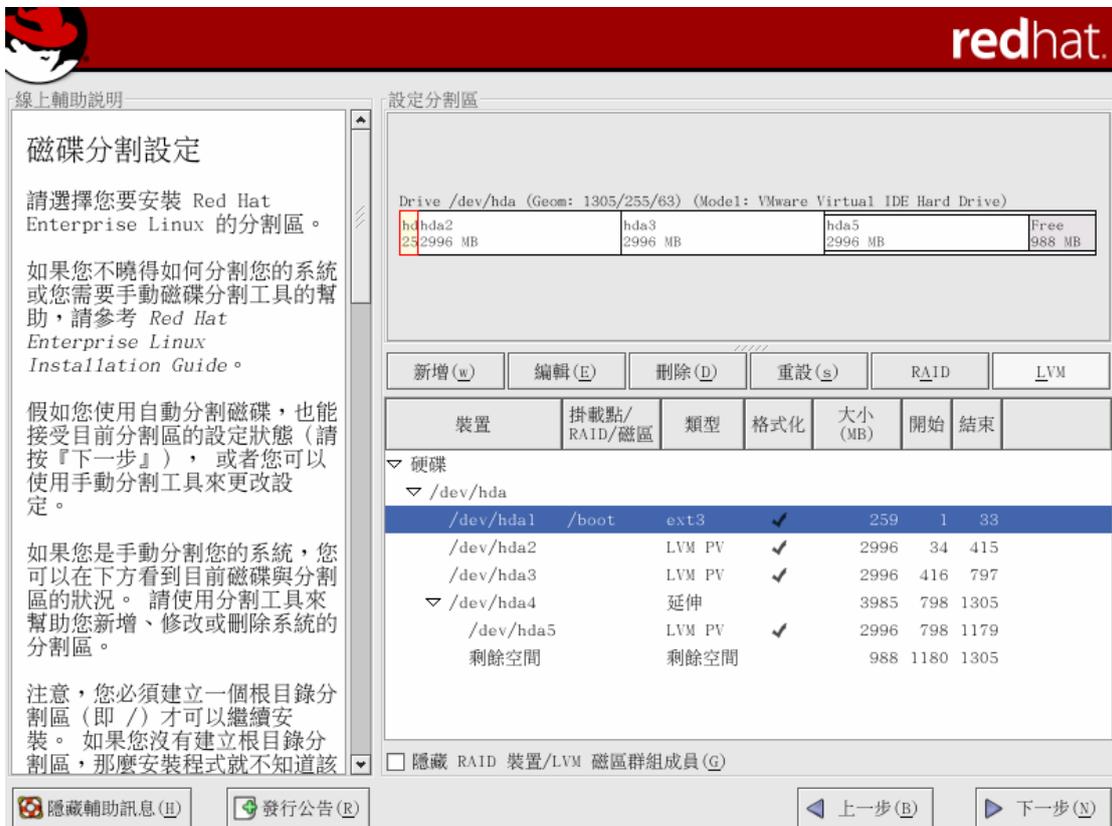


圖 6：建立三個 Physical Volume (PV)

3. 將這三個 Physical Volume (PV) 組合成一個 VG，將此 VG 命名為 rootvg。各位可以把此時的 rootvg 想成是一個虛擬硬碟，地位就如同 hda。所以接下來的工作便是指定每個檔案系統的空間。(圖 7)

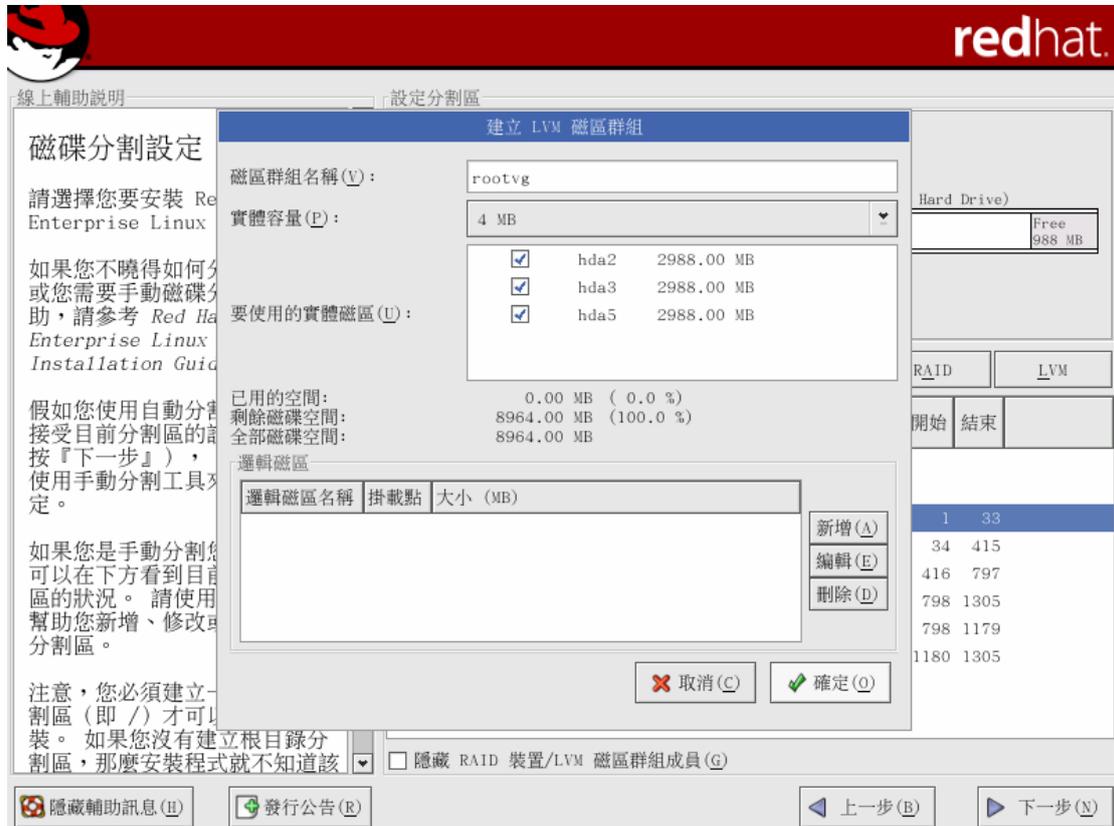


圖 7：將 PV 組合成 VG (rootvg)

4. 在 LVM 機制，當你要建立一個檔案系統時，必須先新增一個 LV (很像以前的分割區 hda#)。新增 LV 時，LVM 會從 VG 中找出未使用的空間組合成你需要的大小，這些空間實際可能對應到任一個硬碟；但使用者不用去在意真正存放在那一個硬碟。也正因這種 Logical 的方式，我們才可將檔案系統的容量放大，因為每個檔案系統對應的已不是從前實體硬碟的分割區 (從第幾個磁柱到第幾個磁柱)，而是 Logical 概念的 LV。只要 VG 還有未用的空間，LVM 就可將其對應到某個 LV，所以 LV 的大小就可增加，藉此達到檔案系統容量可動態放大的功能。

如圖 8~12，筆者指定：

rootlv =6000MB → / 檔案系統

homelv=500MB → /home 檔案系統

swaplv=500MB → SWAP

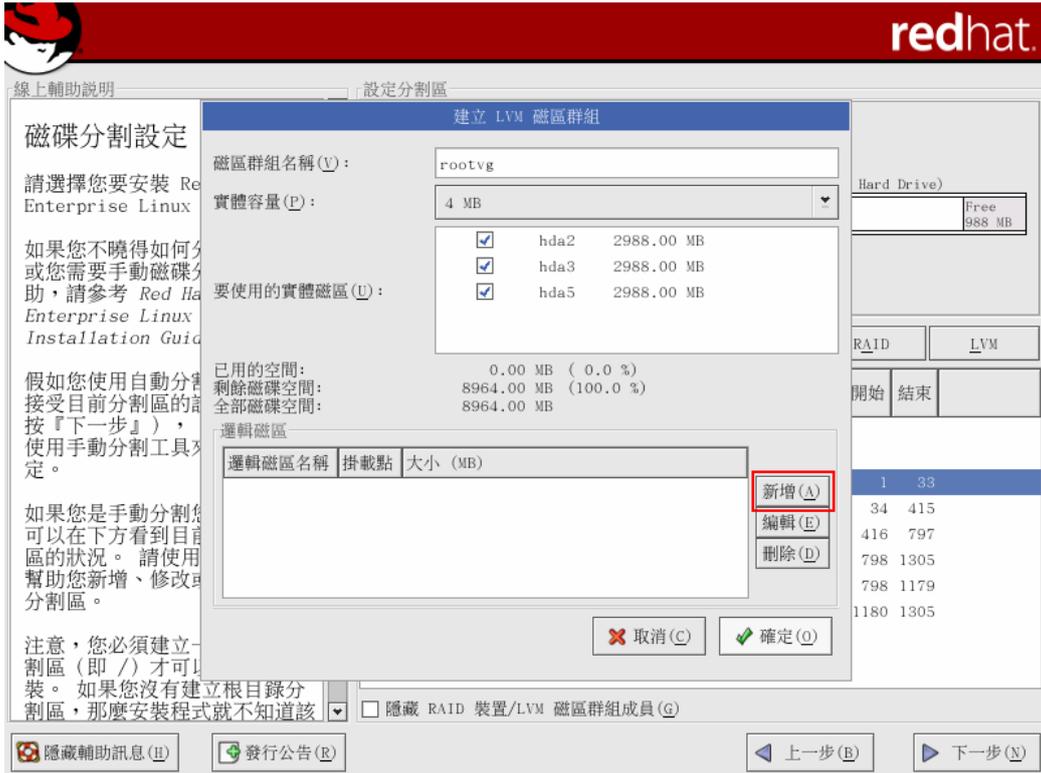


圖 8：新增邏輯磁區 (LV)

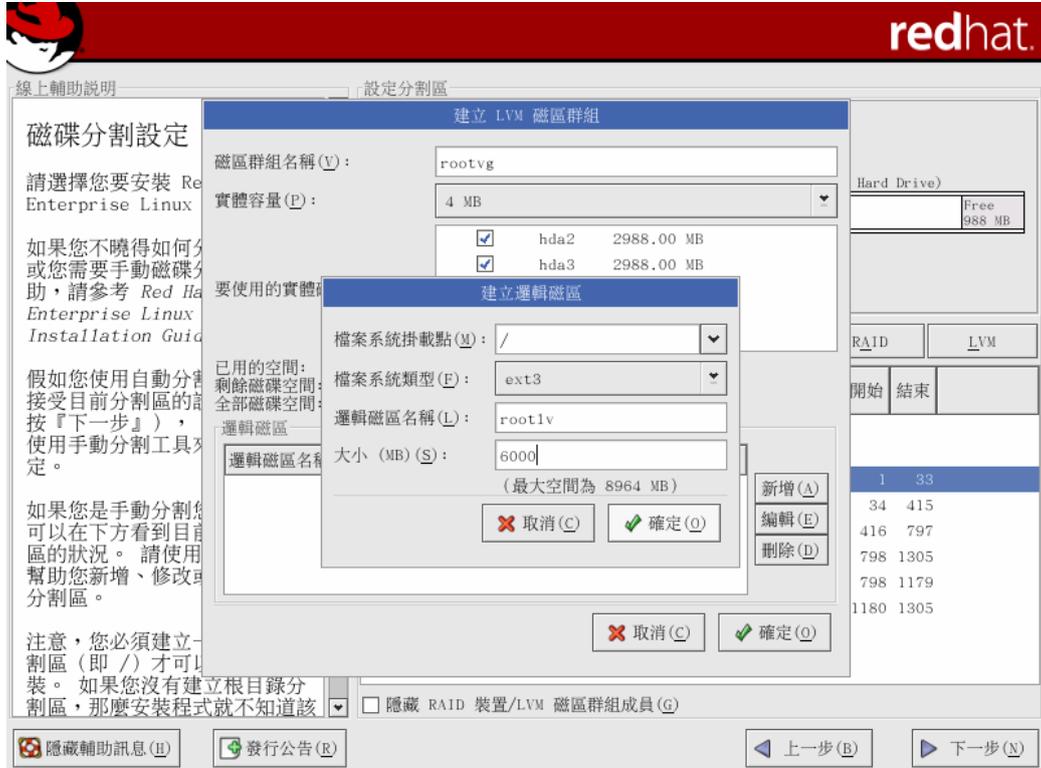


圖 9：新增邏輯磁區名稱 rootlv、其掛載點設定為 /

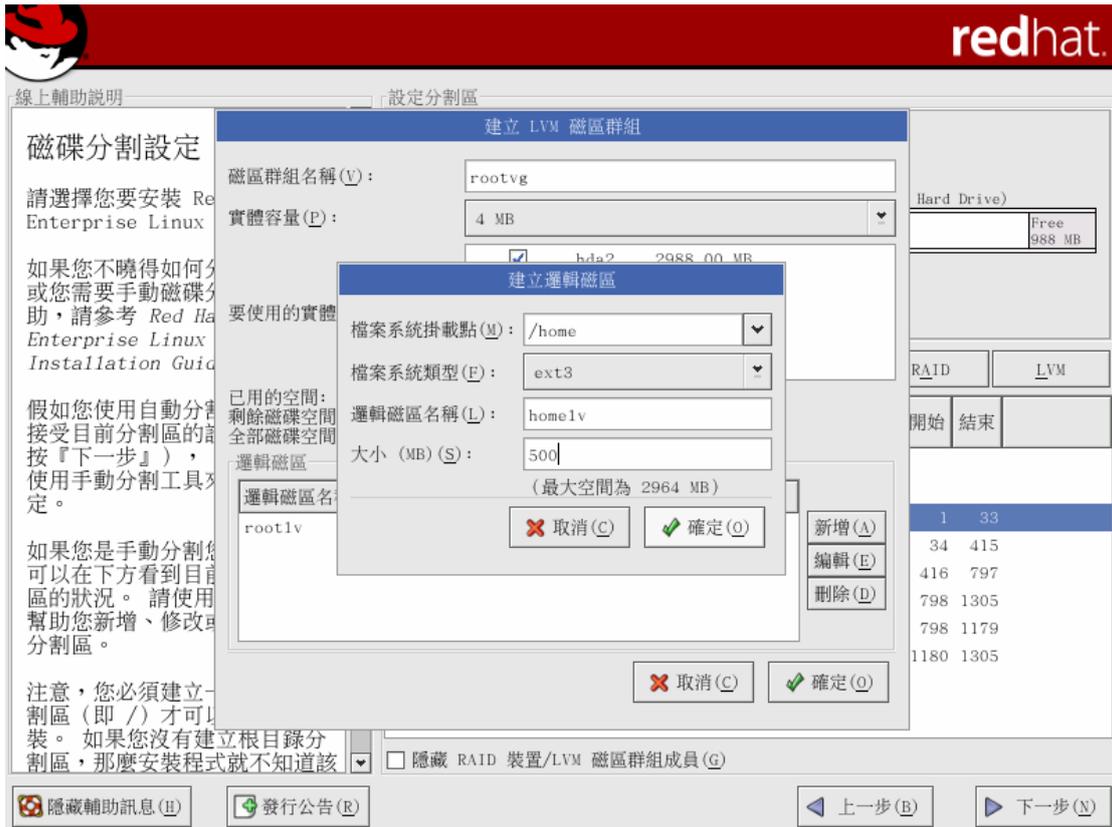


圖 10：新增邏輯磁區名稱 home1v、其掛載點設定為 /home

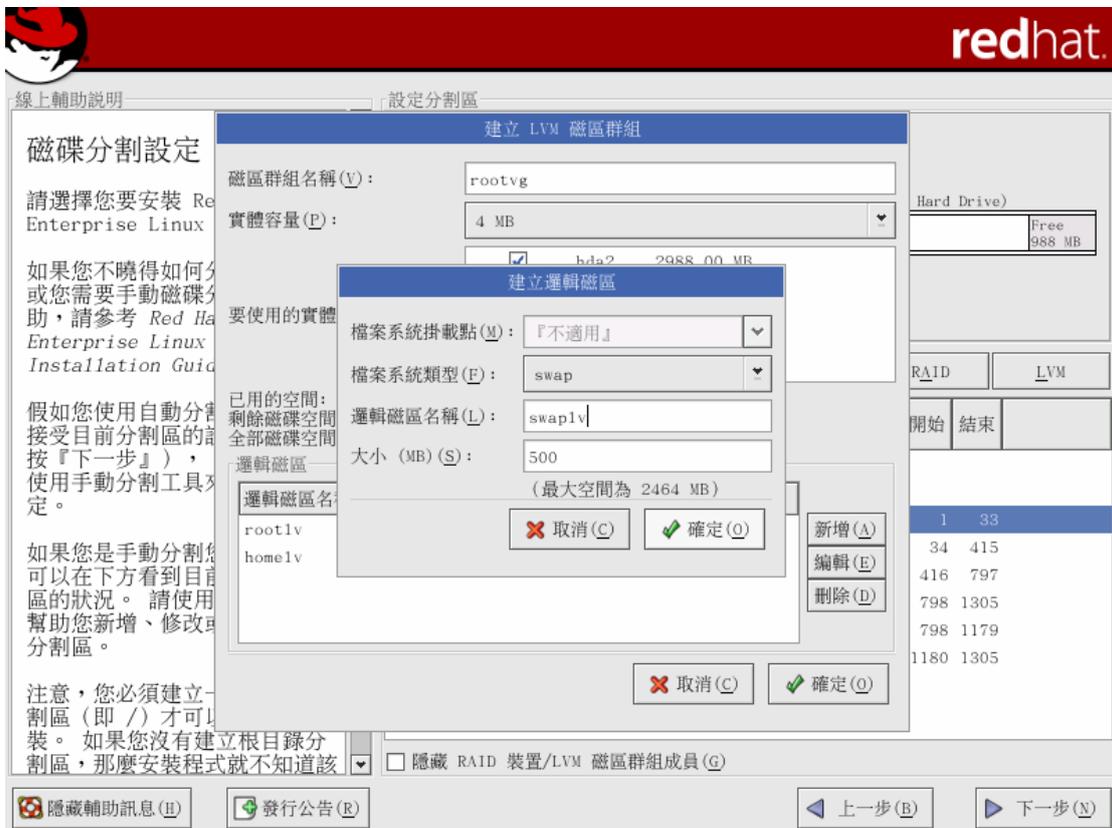


圖 11：新增 swap 邏輯磁區



圖 12：Disk Druid 分割完成圖

安裝完畢開機時各位會看到下列訊息，便代表 LVM 設定成功。

Scanning logical volumes

vgscan – reading all physical volumes (this may take a while...)

vgscan – found inactive volume group “rootvg”

vgscan – “/etc/lvmtab” and “/etc/lvmtab.d” successfully created

.....

Activating logical volumes

vgchange – volume group “rootvg” successfully activated

6.開機後利用 `df -h` 檢查硬碟、LVM、檔案系統的狀況。

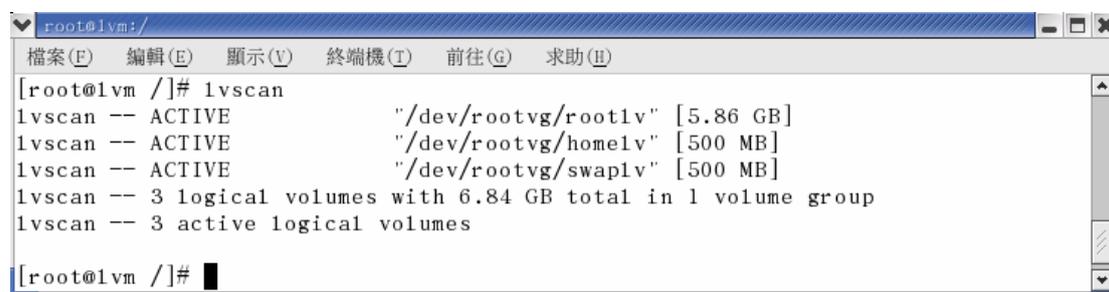
```

root@lvm: /# df -h
檔案系統          容量  已用  可用  已用% 掛載點
/dev/rootvg/rootlv  5.8G  1.9G  3.7G   35%  /
/dev/hda1          251M  14M  225M    6%  /boot
/dev/rootvg/home1v  485M  8.1M  452M    2%  /home
none              125M   0    125M   0%   /dev/shm
/dev/cdrom         626M  626M   0  100%  /mnt/cdrom
root@lvm: /#

```

圖 13：/home 檔案系統容量約 500MB

7. 利用 lvscan 檢查原來 LV 分配情形



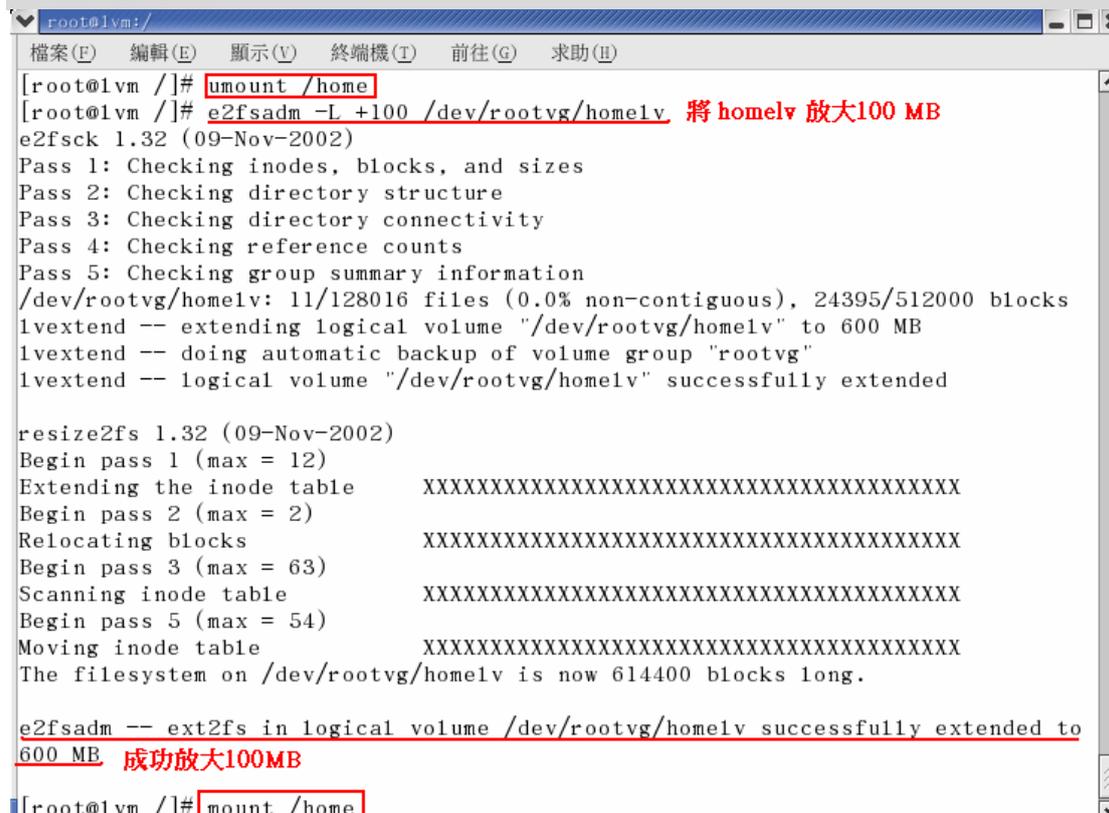
```
[root@lvm ~]# lvscan
lvscan -- ACTIVE          "/dev/rootvg/rootlv" [5.86 GB]
lvscan -- ACTIVE          "/dev/rootvg/homelv" [500 MB]
lvscan -- ACTIVE          "/dev/rootvg/swaplv" [500 MB]
lvscan -- 3 logical volumes with 6.84 GB total in 1 volume group
lvscan -- 3 active logical volumes

[root@lvm ~]#
```

圖 14：利用 lvscan 檢查系統正常啟用的邏輯磁區

8. 動態將 /home 檔案系統放大 100MB

```
#umount /home
#e2fsadm -L+100 /dev/rootvg/homelv
註: -L+100 代表增加 100MB
#mount /home
```



```
[root@lvm ~]# umount /home
[root@lvm ~]# e2fsadm -L +100 /dev/rootvg/homelv 將 homelv 放大100 MB
e2fsck 1.32 (09-Nov-2002)
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/rootvg/homelv: 11/128016 files (0.0% non-contiguous), 24395/512000 blocks
lvextend -- extending logical volume "/dev/rootvg/homelv" to 600 MB
lvextend -- doing automatic backup of volume group "rootvg"
lvextend -- logical volume "/dev/rootvg/homelv" successfully extended

resize2fs 1.32 (09-Nov-2002)
Begin pass 1 (max = 12)
Extending the inode table      XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Begin pass 2 (max = 2)
Relocating blocks             XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Begin pass 3 (max = 63)
Scanning inode table          XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Begin pass 5 (max = 54)
Moving inode table            XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
The filesystem on /dev/rootvg/homelv is now 614400 blocks long.

e2fsadm -- ext2fs in logical volume /dev/rootvg/homelv successfully extended to
600 MB 成功放大100MB

[root@lvm ~]# mount /home
```

圖 15：將/home 檔案系統放大 100MB 過程圖

```
#lvscan → 檢查變更後 LV 分配情形
#df -H 檢查檔案系統使用情形
```

```
root@lvm: /
檔案(E) 編輯(E) 顯示(V) 終端機(T) 前往(G) 求助(H)
[root@lvm /]# df -h
檔案系統 容量 已用 可用 已用% 掛載點
/dev/rootvg/rootlv 5.8G 1.9G 3.7G 35% /
/dev/hdal 251M 14M 225M 6% /boot
none 125M 0 125M 0% /dev/shm
/dev/cdrom 626M 626M 0 100% /mnt/cdrom
/dev/rootvg/homelv 582M 8.3M 544M 2% /home
[root@lvm /]# lvscan
lvscan -- ACTIVE "/dev/rootvg/rootlv" [5.86 GB]
lvscan -- ACTIVE "/dev/rootvg/homelv" [600 MB]
lvscan -- ACTIVE "/dev/rootvg/swaplv" [500 MB]
lvscan -- 3 logical volumes with 6.93 GB total in 1 volume group
lvscan -- 3 active logical volumes
[root@lvm /]#
```

圖 16：/home 檔案系統容量變為 600MB

讀者此時可以檢查原來在 /home 目錄的資料並不會因此而不見。在 AIX 上甚至可以不用做 umount 就將檔案系統放大，筆者認為未來 Linux 也可達到此功能。不過只需簡單「umount → e2fsadm → mount」就可將檔案系統放大，不用似傳統的解決方法需要冗長的停機時間，讓 Linux 更有資格擔任企業關鍵性的伺服器。

2005/06/19 增訂下列文字 ((針對 RHEL 4))

RHEL 4 所含的 LVM 的版本更新至 LVM 2，原本的「umount → e2fsadm → mount」，已由 lvextend 及 ext2online 取代，現已如同 IBMAIX 不用做 umount 的動作便可將檔案系統放大。

```
#umoun /home
#e2fsadm -L +100M /dev/rootvg/homelv
#mount
```

RHEL 4 (LVM 2) 上述步驟由下列指令取代：

```
#lvextend -L +100M /dev/rootvg/homelv
#ext2online /dev/rootvg/homelv
```

4-3：LVM 相關指令

在實例演練 1，筆者利用 Disk Druid 實作 LVM，但 Disk Druid 此工具只有在安裝時才可使用，如果我們在系統安裝完畢後，需要實作 LVM 又該何解呢？接下來筆者將介紹 LVM 指令，並在之後的實例演練利用指令來實作 LVM 機制，其中會告訴讀者如何放大 LV（虛擬分割區）及加大 VG（虛擬硬碟）的大小，還有快速備份 LV 的方法（LV Snapshot）及如何刪除 LVM 中的 VG。

- PV 指令

pvscan 掃描系統中所有 PV 的資訊

```
[root@lvm root]# pvscan
pvscan -- reading all physical volumes (this may take a while...)
pvscan -- ACTIVE   PV "/dev/hda2" of VG "rootvg" [2.92 GB / 0 free]
pvscan -- ACTIVE   PV "/dev/hda3" of VG "rootvg" [2.92 GB / 0 free]
pvscan -- ACTIVE   PV "/dev/hda5" of VG "rootvg" [2.92 GB / 1.83 GB free]
pvscan -- total: 3 [8.78 GB] / in use: 3 [8.78 GB] / in no VG: 0 [0]
```

pvcreate <pv>

建立 Physical Volume

在 Linux LVM 的機制 PV 是由實體的分割區 轉換而成，這個分割區類型必須為 **8e**。當我們建立 PV 時，LVM 機制保留此分割區最前面的區塊用來記錄有關 LVM 的屬性（例如將來加入 VG 後，PE 的大小），此塊空間我們稱為 VGDA（**V**olume **G**roup **D**escriptor **A**rea）。

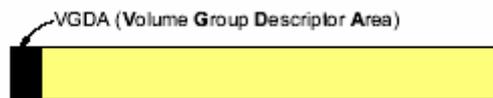


圖 17:VGDA 示意圖

pvmove [-n <lv>] <source pv> [<destination pv>]

搬移 Physical Volume 內資料

pvmove 這個指令並不是搬移 PV，而是搬移 PV 內的資料，其實就是將 PV 中的 PE 資料搬移至同一個 VG 中的另一個 PV 內（註：一定要在同一個 VG 內的 PV），這個指令通常是用於我們要將某個 PV 從 VG 中移除（註：利用 vgreduce 指令）時，先將 PV 中的資料搬移到另一個 PV 中，再進行移除 VG 的動作。

pvdisplay <pv>

顯示 Physical Volume 的屬性

pvdisplay 這個指令就簡單多了，目的是顯示 PV 的資訊，執行訊息如下：

```
# pvdisplay /dev/hda10
```

```
pvdisplay -- "/dev/hda10" is a new physical volume of 54.88 MB
```

- **VG 指令**

vgscan 掃描系統中所有 VG 的資訊

```
[root@lvm root]# vgscan
vgscan -- reading all physical volumes (this may take a while...)
vgscan -- found active volume group "rootvg"
vgscan -- "/etc/lvmtab" and "/etc/lvmtab.d" successfully created
vgscan -- WARNING: This program does not do a VGDA backup of your volume group
```

vgcreate [-s <pe size>] <vg name> <pv> [<pv>...]

建立 Volume Group

我們知道 VG 是由一個或一個以上的 PV 所組成的，當 PV 加入 VG 後會被切成一塊一塊的 PE (預設為 4MB)，所以語法當然就沒有問題。筆者在從事 Linux 教學工作時，常強調先懂原理，再看指令，才不會覺得指令的語法艱澀難懂。

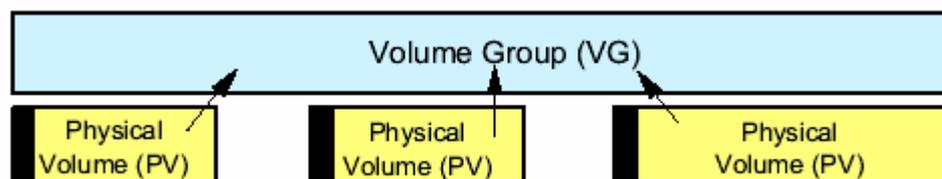


圖 18 : VG 示意圖

```
# vgcreate rootvg /dev/hda10 →用 /dev/hda10 這個 PV 組成 rootvg
vgcreate -- INFO: using default physical extent size 4 MB
vgcreate -- INFO: maximum logical volume size is 255.99 Gigabyte
vgcreate -- doing automatic backup of volume group "rootvg"
vgcreate -- volume group "rootvg" successfully created and activated
```

由上面的 vgcreate 的執行訊息「using **default physical extent size 4 MB**」，便可清楚看到當 PV 加入 VG 後會被切成一塊一塊的 PE(預設為 4MB)。rootvg 後也可接一個以上的 PV，代表一次用多個 PV 組成 rootvg。

vgdisplay [<vg>]

顯示 Volume Group 資訊

此指令會顯示 VG 的大小，其中 PE 的大小，還有此 VG 共有幾塊 PE...等資訊。

vgremove [<vg>]

刪除 Volume Group

刪除 VG 之前，必須讓此 VG 停止作用 (deactive)，目的就是讓所有程式不能再存取此 VG 內的資料。讓 VG 停止作用的指令為

vgchange -a n <vg>。

- **LV 指令**

lvscan 掃描系統中所有 LV 的資訊

```
[root@lvn root]# lvscan
lvscan -- ACTIVE          "/dev/rootvg/rootlv" [5.86 GB]
lvscan -- ACTIVE          "/dev/rootvg/homelv" [600 MB]
lvscan -- ACTIVE          "/dev/rootvg/swaplv" [500 MB]
lvscan -- 3 logical volumes with 6.93 GB total in 1 volume group
lvscan -- 3 active logical volumes
```

lvcreate -L <size> [-n <lv name>] <vg> [<pv>...]

建立 Logical Volume

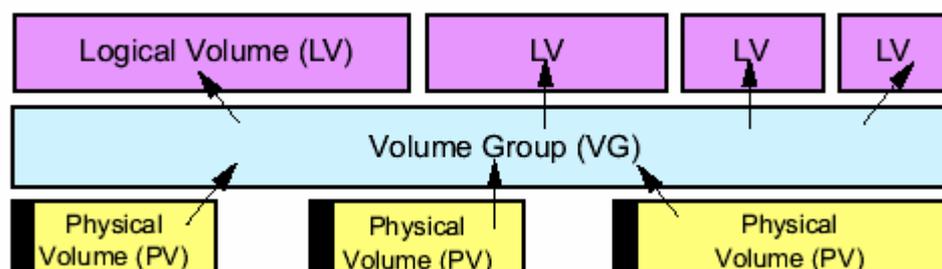


圖 19：LV 示意圖

LV 是 VG 中的一塊空間，就像原來 Linux 中的硬碟中的分割區，所以要建立 LV 就像在硬碟中建立分割區一樣，必須指定要從那個硬碟 (VG) 切割，其大

小為多少。常見的指令用法如下：

#lvcreate -L 512M -n lv01 rootvg

從 rootvg 中切出一個 512MB 的空間的 LV 並指定其名稱為 lv01

#lvcreate -l 32 -n lv01 rootvg

從 rootvg 中挪出 32 個 PE 的空間來建立 LV, 並指定其名稱為 lv01

lvdisplay <lv> [<lv>...]

顯示 Logical Volume 資訊

lvremove <lv> [<lv>...]

刪除 Logical Volume

筆者將上述介紹 LVM 相關的指令整理如下表：

表 4-2：LVM 相關指令

	Physical Volume	Volume Group	Logical Volume
scan (掃描整個系統)	pvscan	vgscan	lvscan
create (建立)	pvcreate	vgcreate	lvcreate
display (顯示)	pvdisplay	vgdisplay	lvdisplay
remove (刪除)		vgremove	lvremove

2005/06/19 增訂下列文字 ((針對 RHEL 4))

由 RHEL 3 的 LVM 檔案系統是 LVM 1 格式，而 RHEL 4 則為 LVM 2 格式，所以 RHEL 4 提供了轉換的指令，讓 RHEL 4 的用戶把原本 LVM 1 的資料轉換至 LVM 2

#vgconvert -M2 <vg 名稱>

實例演練 2 建立 LVM 的檔案系統 (/lvmdata 檔案系統，大小為 30MB)

在實作之前，請讀者回想傳統 Linux 新增檔案系統的作法。假如現在我要新增一個 /data 檔案系統，大小為 30MB，流程應為「新增磁碟分割區」(fdisk) → 「建立檔案系統 (格式化分割區)」(mke2fs) → 「掛載至 /data 目錄」(圖 20)。

在原有 Linux 檔案系統的架構中，一個檔案系統 (假設是 /data 檔案系統) 一定對應某個硬碟的分割區 (/dev/hda#)。/dev/hda# 是實體硬碟從第 X 個磁柱

到 Y 個磁柱，所以 /data 檔案系統實際上是對應到硬碟的某段真實的空間。

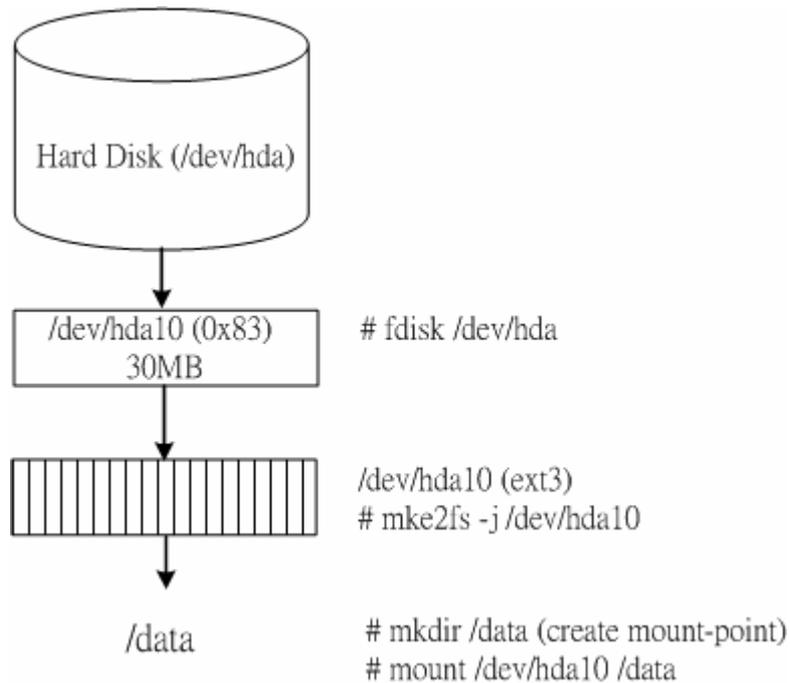


圖 20：傳統 Linux 建立 /data 檔案系統流程圖

步驟及原理：

假設同一個需求，我們利用 LVM 的機制來建立 30 MB /lvm 檔案系統，筆者時常建議實作之前，先想一想整個原理及流程，除了可以加深記憶也比較不會流於只是在敲指令。首先我們得先湊出一個 **VG** (虛擬硬碟)，而 VG 是由 PV 所組成，而 PV 是由實體的分割區所轉換而成的。所以我們第一件事應該是建立 **LVM** 專用的分割區，然後將其轉成 **PV**，再利用它組成 **VG**。(圖 21)

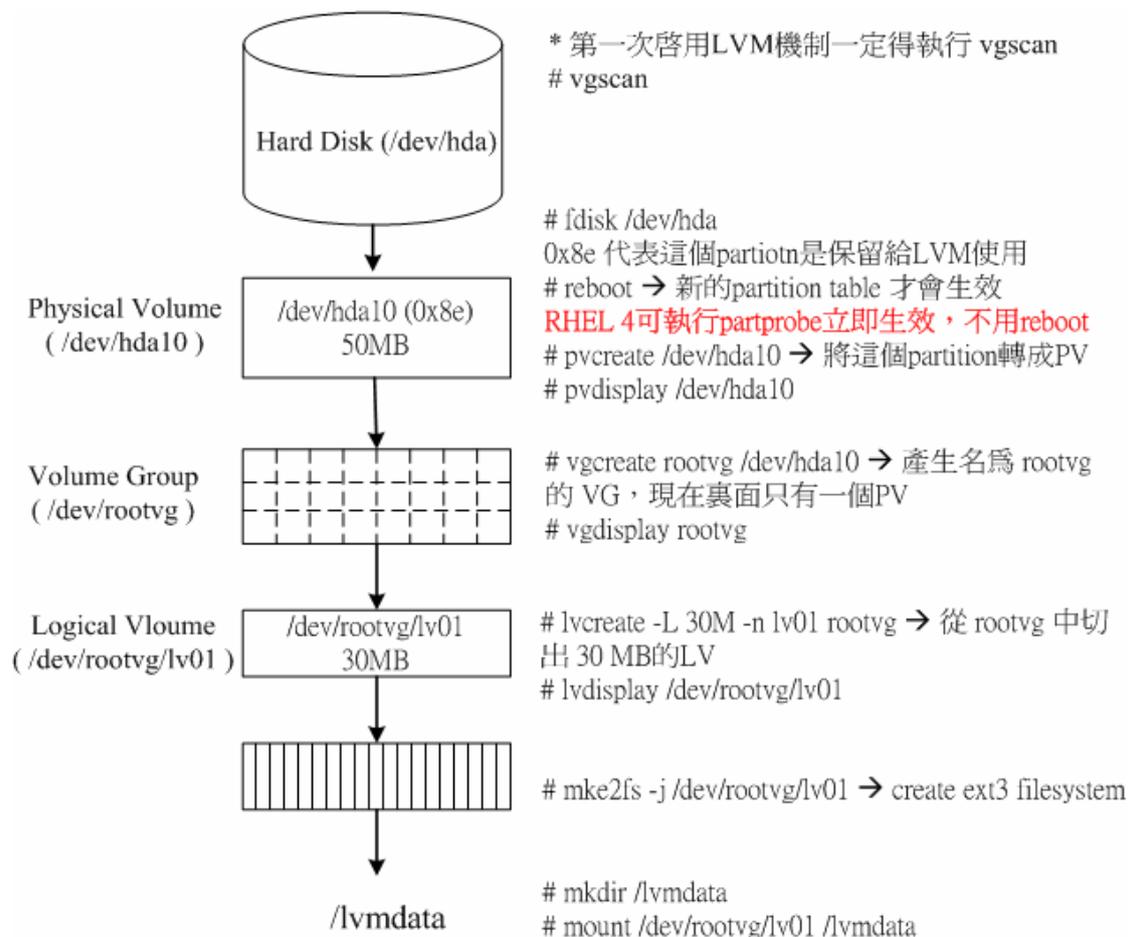


圖 21： Linux LVM create /data 檔案系統流程圖

1. 第一次啟用 LVM 機制一定得執行 `vgscan`

```
# vgscan RHEL 3 啟用 LVM 機制一定得執行 vgscan，RHEL 4 則不需要
vgscan -- reading all physical volumes (this may take a while...)
vgscan -- "/etc/lvmtab" and "/etc/lvmtab.d" successfully created
vgscan -- WARNING: This program does not do a VGDA backup of your
volume group
```

2. 新增 LVM 分割區

`fdisk /dev/hda`

The number of cylinders for this disk is set to 8924.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:

- 1) software that runs at boot time (e.g., old versions of LILO)
- 2) booting and partitioning software from other OSs
(e.g., DOS FDISK, OS/2 FDISK)

```
Command (m for help): n
First cylinder (6698-8924, default 6698):
Using default value 6698
Last cylinder or +size or +sizeM or +sizeK (6698-8924, default 8924): +50M
```

Command (m for help): p

```
Disk /dev/hda: 255 heads, 63 sectors, 8924 cylinders
Units = cylinders of 16065 * 512 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1	*	1	26	208813+	83	Linux
/dev/hda2		27	1301	10241437+	83	Linux
/dev/hda3		1302	2576	10241437+	83	Linux
/dev/hda4		2577	8924	50990310	f	Win95 Ext'd (LBA)
/dev/hda5		2577	2831	2048256	82	Linux swap
.....						
/dev/hda10		6698	6704	56196	83	Linux

Command (m for help):t

```
Command (m for help): t
```

```
分割區 number (1-10): 10
```

Hex code (type L to list codes): 8e

← Changed system type of 分割區 10 to 8e (Linux LVM)

Command (m for help): w

使 **partition table** 生效

#reboot (**RHEL 4** 可執行 **partprobe** 立即生效，不用 **reboot**)

3.將/dev/hda10 轉成 PV

pvcreate /dev/hda10 → 將這個 /dev/hda10 轉成 PV

```
pvcreate -- physical volume "/dev/hda10" successfully created
```

pvdisplay /dev/hda10

```
pvdisplay -- "/dev/hda10" is a new physical volume of 54.88 MB
```

4. 組成虛擬硬碟 rootvg

vgcreate rootvg /dev/hda10 → 用 /dev/hda10 這個 PV 組成 rootvg

rootvg 相當以前 Linux 的一顆硬碟 /dev/hda

```
vgcreate -- INFO: using default physical extent size 4 MB
vgcreate -- INFO: maximum logical volume size is 255.99 Gigabyte
vgcreate -- doing automatic backup of volume group "rootvg"
vgcreate -- volume group "rootvg" successfully created and activated
```

vdisplay rootvg

```
VG Name          rootvg
VG Access        read/write
VG Status        available/resizable
VG #             0
MAX LV           256
Cur LV          1
Open LV          1
MAX LV Size      255.99 GB
Max PV           256
Cur PV          1
Act PV           1
VG Size         48 MB → 這個虛擬硬碟 rootvg 的大小為 48MB
PE Size         4 MB
Total PE         12
Alloc PE / Size  11 / 44 MB
Free PE / Size   1 / 4 MB
VG UUID          kPdjjl-b0fK-1w0D-WagY-Bg6l-VsZk-ReVpPW
```

5. 從虛擬硬碟 rootvg 中切出一塊 LV，名稱 lv01、大小為 30MB。

lvcreate -L 30M -n lv01 rootvg

```
lvcreate -- rounding size up to physical extent boundary
lvcreate -- doing automatic backup of "rootvg"
lvcreate -- logical volume "/dev/rootvg/lv01" successfully created
```

lvdisplay /dev/rootvg/lv01

```
--- Logical volume ---
LV Name          /dev/rootvg/lv01
VG Name          rootvg
LV Write Access  read/write
LV Status        available
LV #             1
# open           1
```

LV Size	32 MB
Current LE	8
Allocated LE	8
Allocation	next free
Read ahead sectors	1024
Block device	58:0

6.將 lv01 格式化，即是建立 ext3 檔案系統的動作。

```
# mke2fs -j /dev/rootvg/lv01
mke2fs 1.32 (09-Nov-2002)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
8192 inodes, 32768 blocks
1638 blocks (5.00%) reserved for the super user
First data block=1
4 block groups
8192 blocks per group, 8192 fragments per group
2048 inodes per group
Superblock backups stored on blocks:
    8193, 24577

Writing inode tables: done
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.
```

7.建立掛載點（mount point）並利用其存取 lv01 的資料（即傳統 linux mount 的觀念）。

```
# mkdir /lvmdata ← 建立 mount-point
# mount /dev/rootvg/lv01 /lvmdata
# df -h
Filesystem          Size  Used Avail Use% Mounted on
/dev/hda2           9.6G  3.2G  5.9G  35% /
```

/dev/hda1	197M	20M	168M	11%	/boot
/dev/hda6	29G	1.6G	25G	6%	/Database
none	440M	0	440M	0%	/dev/shm
/dev/hda3	9.6G	4.0G	5.1G	44%	/usr
/dev/rootvg/lv01	31M	4.1M	25M	14%	/lvmdata

實例演練 3 動態放大檔案系統（將/data 檔案系統由 30 MB 放大為 40MB）

這麼辛苦了建立 /lvmdata 檔案系統，所為何事呢？在原先 Linux 檔案系統的架構下，如果 /lvmdata 不敷使用，即使硬碟還有剩餘的空間，依舊是無法拿來使用，但在 LVM 的架構下可就不一樣了。由以上的流程圖我們知道 rootvg 這類虛擬的硬碟中應該還有 20MB 的剩餘空間，在 LVM 下我們可以把這些空間再分配給 /lvmdata 檔案系統，感覺上很像動態放大 /lvmdata 檔案系統。

RHEL 3 作法如下：

1. umount /lvmdata

```
# umount /lvmdata
```

2. 利用 e2fsadm 指令再新增 10MB 空間給 lv01

```
# e2fsadm -L +10M /dev/rootvg/lv01
e2fsadm -- rounding size up to physical extent boundary
e2fsck 1.27 (8-Mar-2002)
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/rootvg/lv01: 11/8192 files (0.0% non-contiguous), 5166/32768 blocks
lvextend -- extending logical volume "/dev/rootvg/lv01" to 44 MB
lvextend -- doing automatic backup of volume group "rootvg"
lvextend -- logical volume "/dev/rootvg/lv01" successfully extended
resize2fs 1.27 (8-Mar-2002)
Begin pass 1 (max = 2)
Extending the inode table
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Begin pass 3 (max = 4)
Scanning inode table
```

XX

The filesystem on /dev/rootvg/lv01 is now 45056 blocks long.

e2fsadm -- ext2fs in logical volume /dev/rootvg/lv01 successfully extended to 44 MB

放大 lv01 成功

3. 重新 mount /lvmdata 檔案系統

```
# mount /dev/root/lv01 /lvmdata
```

```
# df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/hda2	9.6G	3.2G	5.9G	35%	/
/dev/hda1	197M	20M	168M	11%	/boot

.....
/dev/rootvg/lv01 42M 4.1M 36M 10% /lvmdata

👁 /lvmdata 大小由 30MB 變成 40MB

RHEL 4 作法如下 (2005/06/19 增訂):

RHEL 4 所含的 LVM 的版本更新至 LVM 2，原本的「**umount → e2fsdam → mount**」，已由 **lvextned** 及 **ext2online** 取代。

1. 利用 lvextned 指令再新增 10MB 空間給 lv01

```
#lvextned -L +10M /dev/rootvg/lv01
```

2. 利用 ext2online 讓新增的空間生效

```
#ext2online /dev/rootvg/lv01
```

3. 利用 df -h 檢查空間

```
# df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/hda2	9.6G	3.2G	5.9G	35%	/
/dev/hda1	197M	20M	168M	11%	/boot

.....
/dev/rootvg/lv01 42M 4.1M 36M 10% /lvmdata

👁 /lvmdata 大小由 30MB 變成 40MB

實例演練 4 動態加大虛擬 HD (Volume Group) 大小

問題又來，假設 rootvg 這顆虛擬硬碟已沒有剩餘的空間，怎麼辦？沒關係我們可以放大 rootvg 的空間。怎麼放大 rootvg 的空間呢？只要再新增一個 PV 並將其加入 rootvg，那麼 rootvg 空間就會加大，感覺上很像動態放大硬碟的空間。

1. 再新增一個 50MB 分割區並轉成 PV

```
# fdisk /dev/hda → add new 0x8e 分割區 /dev/hda11 (50MB)
#reboot (RHEL 3) 或 partprobe (RHEL 4)
# pvcreate /dev/hda11
```

2. 並把它加入 rootvg，撐大 rootvg 這顆虛擬硬碟的大小

```
# vextend rootvg /dev/hda11
vextend -- INFO: maximum logical volume size is 255.99 Gigabyte
vextend -- doing automatic backup of volume group "rootvg"
vextend -- volume group "rootvg" successfully extended ← 成功地放大 rootvg
```

```
# vgdisplay rootvg
VG Name                rootvg
VG Access               read/write
VG Status               available/resizable
VG #                    0
MAX LV                  256
Cur LV                 1
Open LV                 1
MAX LV Size             255.99 GB
Max PV                  256
Cur PV                 2
Act PV                  2
VG Size                 96 MB ← 變成 96 MB
PE Size                 4 MB
Total PE                24
Alloc PE / Size         11 / 44 MB
Free PE / Size          13 / 52 MB
VG UUID                 kPdjjl-b0fK-1w0D-WagY-Bg6l-VsZk-ReVpPW
```

實例演練 5 刪除 VG

最後我們演練將之前所建立的 rootvg 刪除，刪除 VG 時，必須先將 VG 內的 LV 先移除，並將 VG 停止作用(deactivate),才能此 VG 刪除。

1. 刪除 LV

```
#umount /lvmdata
#lvremove /dev/rootvg/lv01
lvremove /dev/rootvg/lv01
lvremove -- do you really want to remove "/dev/rootvg/lv01"? [y/n]: y
lvremove -- doing automatic backup of volume group "rootvg"
lvremove -- logical volume "/dev/rootvg/lv01" successfully removed
```

2. 將 VG 停止作用(deactivate)

```
# vgchange -a n rootvg
vgchange -- volume group "rootvg" successfully deactivated
```

3. 刪除 VG

```
# vgremove rootvg
vgremove -- volume group "rootvg" successfully removed
```