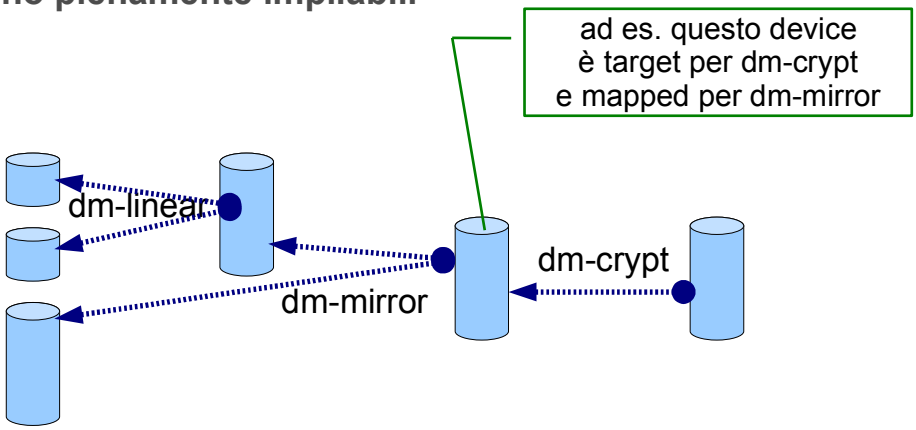
Tecniche per la salvaguardia della disponibilità ed integrità dei sistemi di elaborazione e delle informazioni

2. flessibilità dei sistemi di storage

Marco Prandini
Università di Bologna

Device mapper

- Il device mapper (dm) è un modulo del kernel che, in termini generali, mappa un block device su altri secondo una data politica.
  - un *target driver*
    - espone un block device virtuale (*mapped device*)
    - è associato a dei block device virtuali o reali sottostanti (*target devices*)
  - in userspace si definisce la politica per *mappare i singoli blocchi del mapped device sui blocchi dei target*
  - i device sono pienamente impilabili

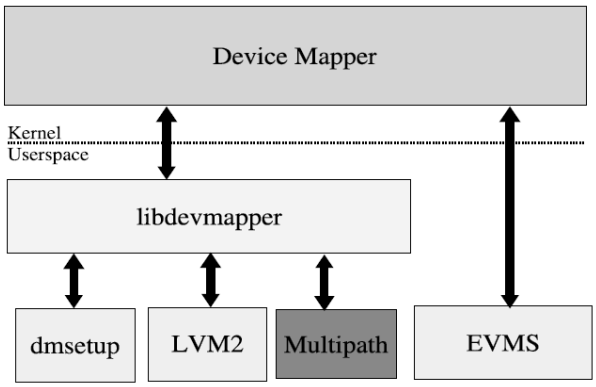


Device mapper

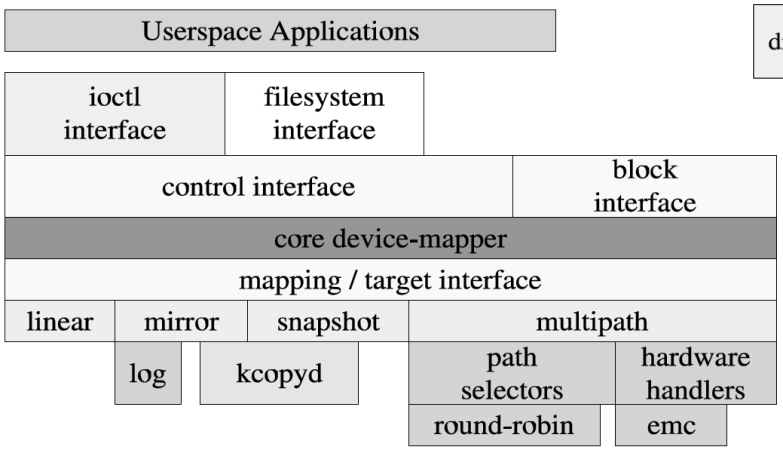
Essenzialmente: tramite tool userspace

- **si scelgono i target**
- **si mappano i blocchi dei mapped sui target**

Userspace Architecture



Device Mapper Kernel Architecture



... poi i moduli kernel si occupano di smistare i dati

- **secondo un certo algoritmo**
- **sui target configurati**
- **seguendo le mappe**

Device mapper

- Il tool userspace di base è dmsetup; dati in ingresso
  - un target driver
  - un elenco di target device
  - una tabella di mapping compatibile col driver

  configura un mapped device su cui si può leggere e scrivere, provocando in realtà letture e scritture sui target device in accordo alla tabella di mapping

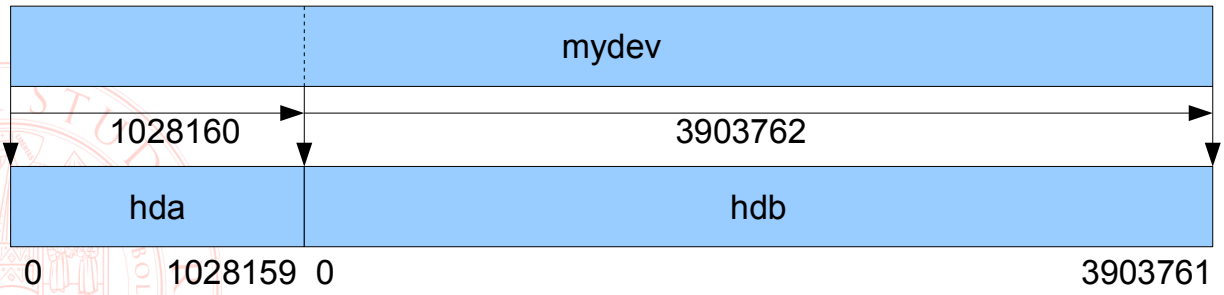
  e naturalmente può deconfigurarlo, dare informazioni, ...
- I target driver attualmente disponibili sono
  - *linear*: concatenazione di target device
  - *striped*: distribuzione di blocchi attraverso i target device
  - *mirror*: replicazione di blocchi sui target device
  - *multipath*: selezione del percorso di accesso ai target device
  - *snapshot*: stacking di target device con accesso copy-on-write
  - *crypt*: cifratura e decifrazione trasparente dei dati sul target dev.
  - *error:* simulazione di errori ai fini di testing

Device mapper

- Esempio 1
  (tabelle da preparare per il comando **dmsetup create mydev <tabella>**, che crea il mapped device /dev/mapper/mydev – supponiamo di disporre di un disco hda di 1028160 settori ed un disco hdb di 3903762 settori)
- unire i dischi hda ed hdb insieme uno di seguito all'altro

```
0          1028160 linear     /dev/hda      0
1028160    3903762 linear     /dev/hdb      0
```

mydev start | num sectors | target driver | target device | target dev. start sector

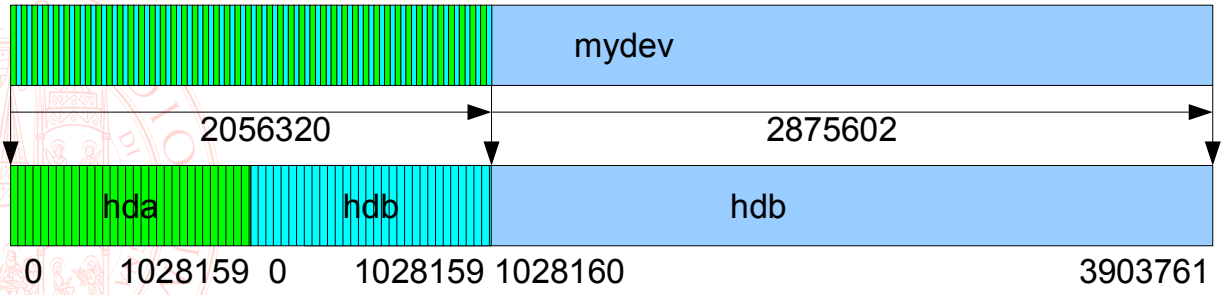


Device mapper

- Esempio 2
  (tabelle da preparare per il comando **dmsetup create mydev <tabella>**, che crea il mapped device /dev/mapper/mydev – supponiamo di disporre di un disco hda di 1028160 settori ed un disco hdb di 3903762 settori)
- distribuire i dati in striping tra hda ed hdb ed aggiungere in coda lo spazio residuo di hdb

num stripes | chunk size | =1028160*2

```
0          2056320 striped 2 32 /dev/hda 0 /dev/hdb 0
2056320    2875602 linear /dev/hdb 1028160
```

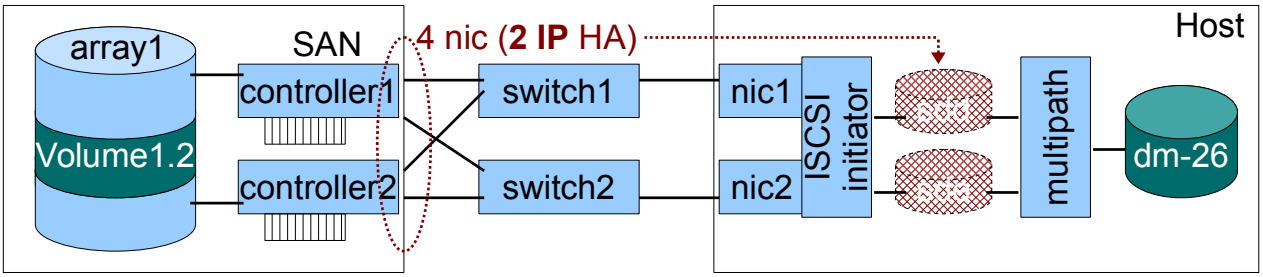


Device mapper – target applications

- Sebbene sia possibile utilizzare dmsetup direttamente anche per fini complessi, esistono alcune applicazioni di più alto livello che permettono di gestire in modo potente task di particolare utilità
- Per l'alta disponibilità:
  - dmraid (specifico per l'utilizzo di controller ATARAID)
  - *dm-multipath*
- Per sicurezza e flessibilità d'uso dei device
  - dmcrypt
  - *LVM2*

Device mapper – multipath

- Percorsi multipli di accesso allo storage:
  - Il kernel riconosce ogni path come un block device indipendente
  - Il comando **`multipath`** interroga i device, riconosce che sono "incarnazioni" della stessa periferica, e genera un block device virtuale
  - Il demone **`multipathd`** reindirizza il traffico in caso di path failure




```
# multipath -v 2 -l
mpath5 (36006048c6f2a4f0ca673c9d09e909be5) dm-26 EMC,Celerra
[size=1.0T][features=0][hwhandler=0][rw]
\_ round-robin 0 [prio=0][active]
 \_ 24:0:0:0 sdd 8:48  [active][undef]
\_ round-robin 0 [prio=0][enabled]
 \_ 25:0:0:0 sde 8:64  [active][undef]
```

Device virtuale

Politica di accesso

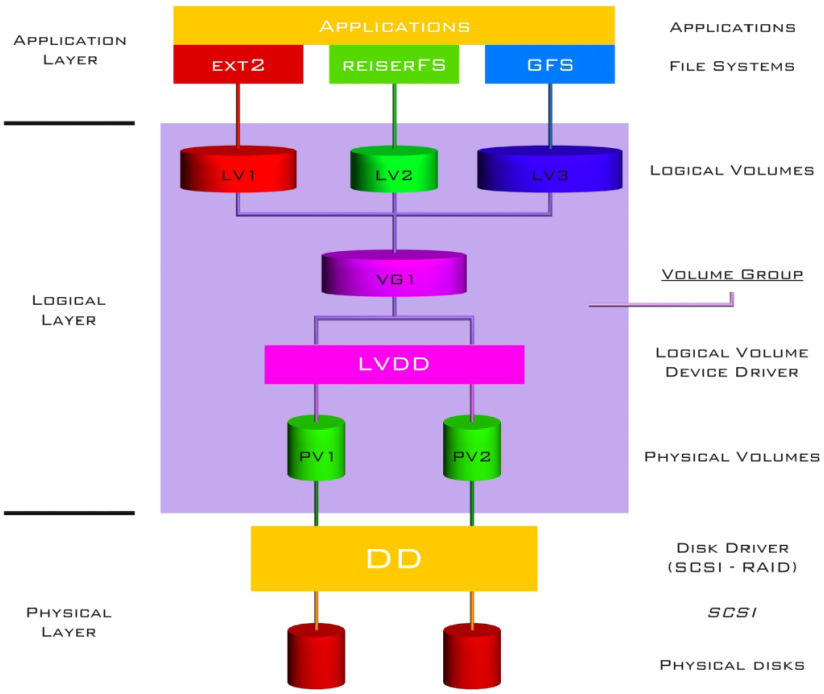
Device reali

Device mapper – multipath

- Esempi di aspetti parametrizzabili (`/etc/multipath.conf`)
  - Come determinare lo stato di salute di un path
    - Con che frequenza testare
    - Con che metodo (lettura disco, driver hw specifici, ...)
  - Come smistare il traffico in condizioni normali e di path failure
    - Politiche di bilanciamento
    - Politiche di priorità
    - Accodamento in caso di total failure vs. segnalazione errore
  - Come gestire la riattivazione di un path
    - Failback immediato o manuale
  - Quali device (non) testare per verificare se rappresentano path alternativi verso un unico storage

Device mapper – LVM2

- Tutti i sistemi visti
  - mettono a disposizione un disco fisico
  - il disco viene rigidamente partizionato in unità logiche
- Manca la cosa forse più semplice: astrarre la gestione dello spazio disponibile
- LVM (Logical Volume Manager)
  - vede i device come PV
  - raggruppa i PV in VG
  - preleva flessibilmente spazio dai VG per definire LV

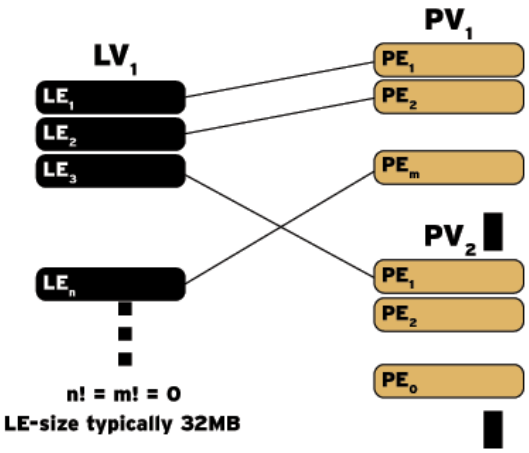


Device mapper – LVM2: elementi

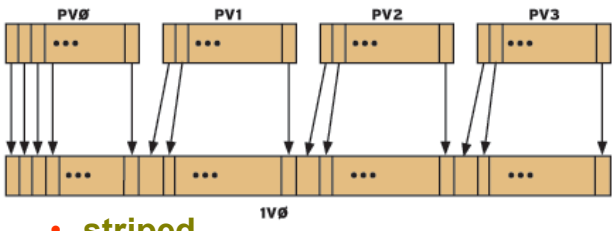
- Physical Volume (PV)
  - l'astrazione LVM dei dischi fissi o partizioni o loopback device o md device
  - sono semplicemente i block device inizializzati con metadati nel settore Volume Group Descriptor Area (VGDA)
- Volume Group (VG)
  - la più alta astrazione di volume ottenibile con LVM.
  - unisce sotto di se una collezione di Logical Volume (LV) e Physical Volume (PV) in un unica unità amministrativa
  - visto dal sistema come un grande disco logico, scomponibile in più partizioni, anch'esse logiche, come i Logical Volume (LV) formati dallo spazio fisico dei PV che compongono il VG.
- Logical Volume (LV)
  - estende il concetto di partizione standard
  - il sistema riconosce i LV come normali block device locali
  - sui LV vengono creati i file system

Device mapper – LVM2: mappature

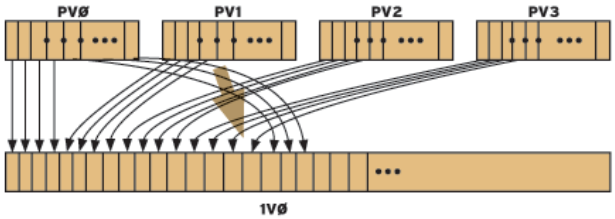
- I VG sono visti come collezioni di Physical Extent (PE)
  - i PE sono materialmente sui PV
- I LV sono visti come sequenze di Logical Extent (LE)
  - i LE sono in corrispondenza 1:1 con i PE



  - la mappatura può essere di vari tipi, es:
    - lineare



    - striped

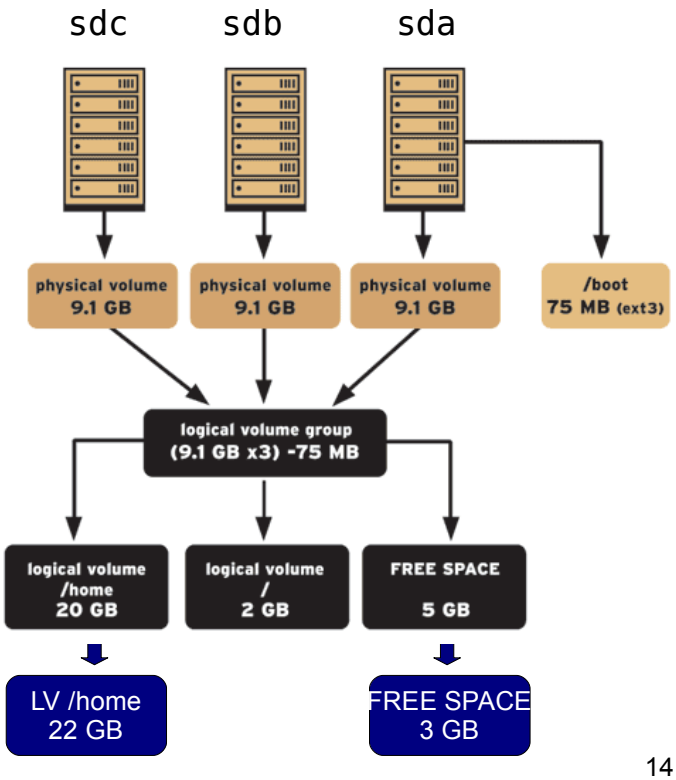


Device mapper – LVM2: vantaggi

- LVM è basato su dm e quindi può avvalersi delle sue funzioni
  - mappature orientate alla ridondanza o alle prestazioni
  - snapshot
- LVM ha un'interfaccia userspace molto più ricca di dmsetup
  - implementa tali funzioni, che richiederebbero lunghe ed accurate sequenze di comandi, in modo semplice e robusto
- LVM presenta ai gestori dei filesystem device
  - che possono essere ridimensionati online
  - i cui guasti fisici (se tollerabili) possono essere riparati trasparentemente
    - è possibile sostituire i PV che compongono un LV mentre questo funziona

Device mapper – LVM2: creazione di volumi

```
fdisk → tag sda2 0x8e
pvcreate /dev/sda2
pvcreate /dev/sdb
pvcreate /dev/sdc
vgcreate vg1 /dev/sda2 /dev/sdb
lvcreate -n lv1 --size 2G vg1
vgextend vg1 /dev/sdc
lvcreate -n lv2 --size 20G vg1
mkfs.ext3 /dev/mapper/vg1-lv1
mkfs.ext3 /dev/mapper/vg1-lv2
mount /dev/mapper/vg1-lv1 /
mount /dev/mapper/vg1-lv2 /home
mount /dev/sda1 /boot
lvextend -L+2G /dev/vg1/lv2
resize2fs /dev/vg1/lv2
```



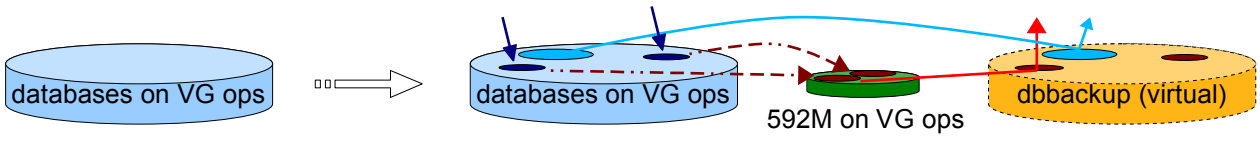
Device mapper – LVM2: altri comandi utili

- Visualizzare le risorse
  - [pv|vg|lv]display
- Rimuovere risorse
  - [pv|vg|lv]remove
- Scansione dei dischi alla ricerca di elementi LVM
  - [pv|vg|lv]scan
- Operazioni sui LV
  - lvconvert (per snapshot o gestione mirror)
  - lvresize

Device mapper – LVM2: snapshot

- Snapshot: creazione di una copia virtuale di un LV che fotografa il suo contenuto in quell'istante
  - ad es. per fare backup senza problemi di inconsistenza dei file

```
lvcreate -L592M -s -n dbbackup /dev/ops/databases
```



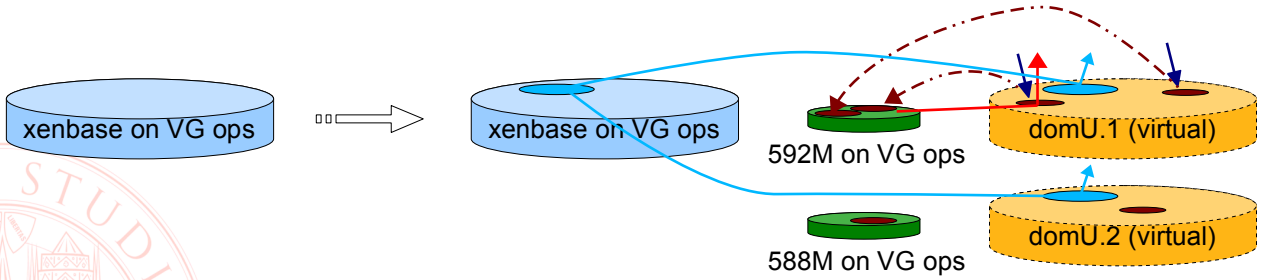
ogni scrittura su databases provoca il salvataggio del settore originale sullo spazio di snapshot
la lettura su dbbackup di un settore modificato viene fatta dallo snapshot
la lettura su dbbackup di un settore non modificato viene fatta da databases

- al termine, si prosegue ad operare semplicemente rimuovendo dbbackup
- è possibile attivare in sequenza vari snapshot
  - si possono implementare politiche di rollback point-in-time

Device mapper – LVM2: snapshot

- è possibile accedere in RW agli snapshot
  - uso opposto al precedente
    - non tocco l'originale (potrebbe essere RO, utile per creare tanti filesystem poco differenziati su una base comune)
    - le scritture sullo snapshot sono allocate sullo "spazio a perdere" (quindi in questo caso la rimozione è un rollback allo stato iniziale)

```
lvcreate -L592M -s -n domU.1 /dev/ops/xenbase
lvcreate -L588M -s -n domU.2 /dev/ops/xenbase
```



ogni scrittura su domU.1 viene fatta sullo spazio di snapshot

la lettura su domU.1 di un settore modificato viene fatta dallo snapshot

la lettura su domU.1 di un settore non modificato viene fatta da xenbase