Filesystem e Dischi

Problemi e soluzioni

Federico Amedeo Izzo

federico.izzo42@gmail.com

POLITECNICO OPEN
unix LABS
Come hack with us.

Benvenuti

Queste slides sono disponibili su

`filesystem.izzo.ovh`

Archiviazione

Argomenti principali:

Argomenti principali:

- Disk failure e silent data corruption

Argomenti principali:

- Disk failure e silent data corruption
- Privacy dei dati

Argomenti principali:

- Disk failure e silent data corruption
- Privacy dei dati
- Volume management

Argomenti principali:

- Disk failure e silent data corruption
- Privacy dei dati
- Volume management
- Snapshot / Backup

Disk failure: gli hard disks si rompono

Se un hard disk si rompe, i dati al suo interno vengono persi (a meno di un backup)

Disk failure: gli hard disks si rompono

Se un hard disk si rompe, i dati al suo interno vengono persi (a meno di un backup)

Una soluzione è il RAID

Redundant Array of Independent Disks

Divide i dati su più dischi permettendo:

- Sopravvivenza alla rottura di uno o più dischi
- Aumento di prestazioni rispetto ad un disco singolo

RAID 0

Divide i dati su più dischi

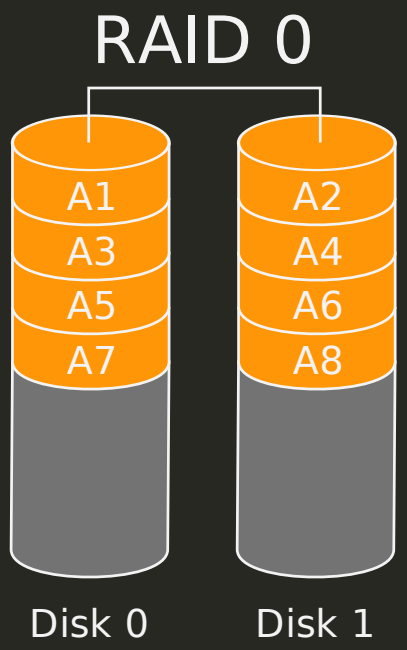
effettuando lo *striping*

pro: alte prestazioni su R / W

contro: affidabilità **peggiore**

di un disco singolo,

non fault tolerant



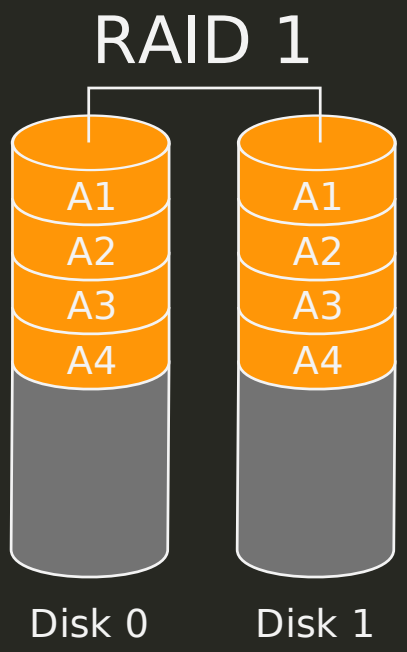
RAID 1

Data *mirroring*

tra due o più dischi

pro: buona velocità lettura, fault tolerant

contro: velocità scrittura pari al disco più lento



RAID 5

Data *striping*

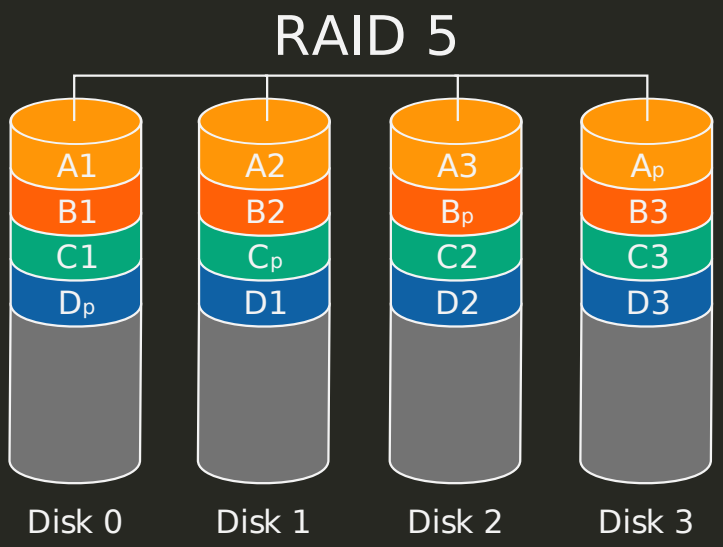
with distributed *parity*

pro: lettura veloce,

fault tolerant

contro: scrittura lenta

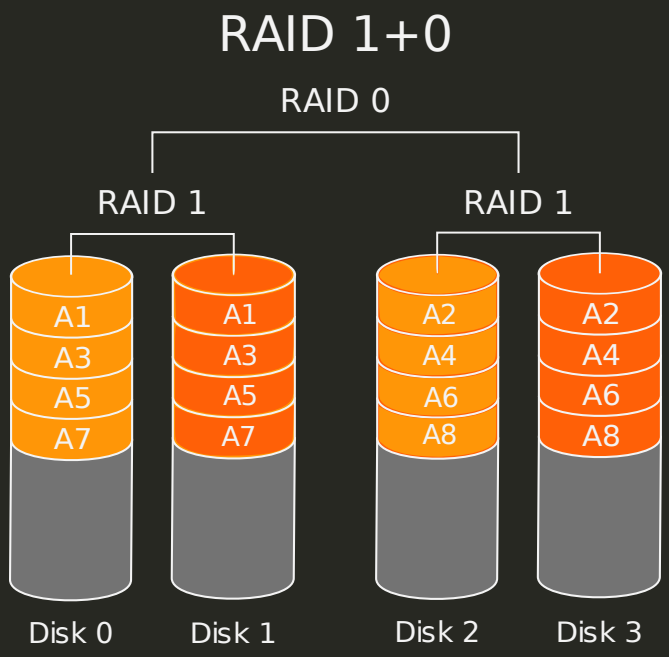
per calcolo XOR



RAID 10 (1+0)

Data *striping*

between *mirrored* disks



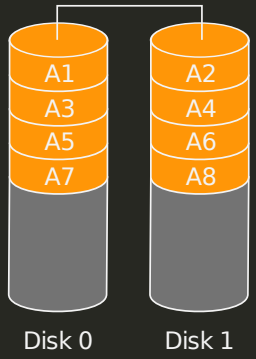
pro: Performance ottime,

fault tolerant

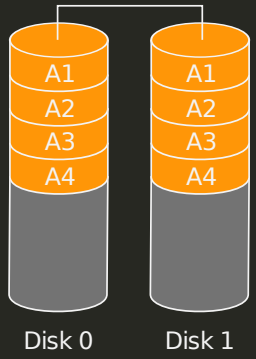
contro: richiede 4 dischi,

solo metà della capacità è utilizzabile

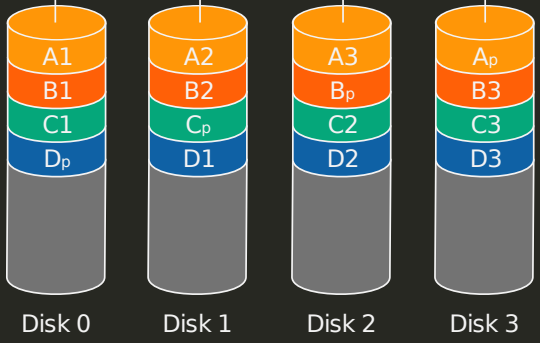
RAID 0



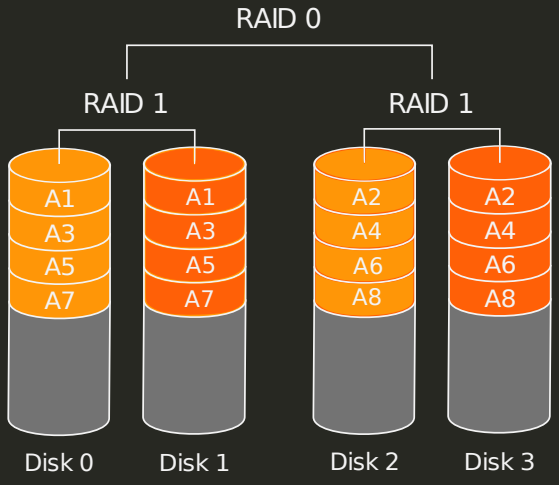
RAID 1



RAID 5



RAID 1+0



Inconvenienti:

RAID può rimediare alla rottura di un disco ma non protegge da errori R/W o dati corrotti

Silent data corruption:

Il cambiamento anche di un solo bit può portare a corruzione dei dati, che noteremo solo al momento del loro utilizzo

Se non eliminato, un errore può venire replicato in un backup

Soluzioni:

- **data scrubbing:**

  controllo periodico del disco che segnala *bad sectors* e inconsistenze con i parity bit o tra le copie

  Non sempre è in grado di decidere quale di due dati discordanti è quello sano

- **checksum dei blocchi**:

  filesystem come ZFS o Btrfs

ZFS

Creato da Sun Microsystems nel 2005,
Distribuito con licenza incompatibile con GPL,
problemi di integrazione nel kernel linux

Btrfs

Sviluppato da Oracle nel 2007, attualmente
contribuiscono Facebook, Intel, Red Hat

Entrambi dispongono di features avanzate:

Entrambi dispongono di features avanzate:

- **CopyOnWrite**

  Quando si copia un file, sul disco vengono aggiunti solo i dati che lo distinguono dal primo

Entrambi dispongono di features avanzate:

- **CopyOnWrite**

  Quando si copia un file, sul disco vengono aggiunti solo i dati che lo distinguono dal primo

- **Deduplication**

  Evita di scrivere copie dello stesso dato (ZFS) o scansiona il disco per eliminare duplicati (Btrfs)

- **Data** & **Metadata checksums**

  Viene salvato un checksum(hash) per ogni blocco di dati e metadati,

  Permette di rivelare *silent data corruption*

- **Data** & **Metadata checksums**

  Viene salvato un checksum(hash) per ogni blocco di dati e metadati,

  Permette di rivelare *silent data corruption*

- **Funzionalità RAID integrate**

  ZFS offre RAID-Z (simile a RAID5), Btrfs può creare e gestire tutti i tipi di RAID visti finora

  Se un checksum fallisce, ZFS e Btrfs sfruttano il RAID per recuperare in modo trasparente i dati

- **Volume management**

  Gestisce le partizioni in modo virtuale, permette di modificarle in modo semplice e meno rischioso

- **Volume management**

  Gestisce le partizioni in modo virtuale, permette di modificarle in modo semplice e meno rischioso

- **Snapshots**

  Possibilità di creare un'immagine di un intero volume ad un dato istante (COW), e di ripristinarla o utilizzarne i file

Features **ZFS** e **Btrfs**, domande?

- CopyOnWrite
- Deduplication
- Data & Metadata checksums
- Funzionalità RAID integrate
- Volume management
- Snapshots

Uno sguardo alla **sicurezza**:

I dati su un disco non cifrato sono accessibili avviando un altro sistema sul PC (es: usb linux)

LUKS

Standard per la cifratura dei dischi su linux

- Viene richiesta una passphrase all'avvio del pc per sbloccare il disco
- Il disco è normalmente utilizzabile una volta sbloccato

Lo spazio libero del disco deve essere indistinguibile da quello occupato da LUKS, inoltre serve eliminare tracce dei vecchi dati non cifrati

Per questo è utile scrivere dati casuali sull'intero disco.

/dev/sdX è il disco o partizione di destinazione (fare attenzione)

```
# dd if=/dev/urandom of=/dev/sdX bs=4096
```

LVM

Logical Volume Management: permette di gestire le partizioni ad un livello di astrazione superiore.

- Volume management e snapshot (COW)
- Rinominare e raggruppare partizioni
- Disk striping
- Mirrored volumes

LVM è superfluo con Btrfs o ZFS, in quanto integrano già queste features

LVM è strutturato in:

- Physical Volume (PV):

  Disco o partizione utilizzabile da LVM

- Volume Group (VG):

  Disco virtuale formato da uno o più PV (anche su dischi diversi)

- Logical Volume (LV):

  Partizioni virtuali all'interno di un VG

Snapshots/Backup

Uno **snapshot** è un' immagine di un intero volume, permette di ripristinare il sistema ad uno stato precedente

Un **backup** fornisce una copia di dati importanti ma non copre l'intero sistema

RAID is not backup

Come abbiamo visto RAID protegge solo da alcuni fattori,

Un **backup** è essenziale in ogni caso!