Architettura e caratteristiche dello storage in INFN-CC

Workshop CCR • La Biodola 3-7/06/2019

Marica Antonacci, Stefano Stalio per INFN-CC

Storage affidabile ed a basso costo in INFN-CC

Workshop CCR • La Biodola 3-7/06/2019

Marica Antonacci, Stefano Stalio per INFN-CC

Hardware layout

2 Storage node per sede
12 HDD per server (6TB o 8TB)
6 SDD per server (500GB)

NO RAID
OBJECT STORAGE: SWIFT, CEPH

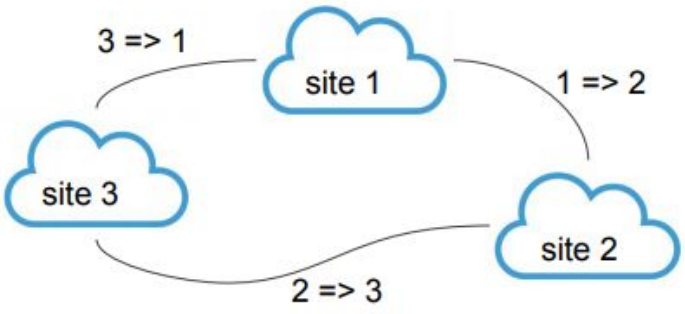
Nel 2019 previsto riempimento degli chassis (+6 dischi per storage node)

Caratteristiche dello storage in INFN-CC

- NO HW RAID
- object storage (**CEPH**, **Swift**)
- metadati su SSD
- dati su HDD (o SSD, su CEPH)
- dati replicati (2x o 3x) o “**erasure coded**”
- CEPH e Swift implementati sullo stesso HW, le risorse allocate sono facilmente migrabili da un sistema all’altro, in caso di necessità

CEPH in INFN-CC

- storage backend per tutti i servizi più importanti di INFN-CC
- 3 cluster indipendenti
- accesso posix attraverso volumi montati dalle vm
- alcuni pool di block device replicati su un’altra sede (RBD mirroring) per DR



CEPH in INFN-CC

Use cases

- system disks per VM (nova)
- volumi posix da montare sulle VM (cinder)
  - l’utente può scegliere il tipo di volume da usare (default, SSD, con replica remota,...)
- in futuro possibili anche gli use case di Swift (radosgw)

Reducing the storage costs

- Replica 3 is a typical configuration
  - ~33% of the raw capacity is usable!
- Ceph **Erasure Coding** allows to achieve greater usable capacity

```
n = k + m  where ,
k  =  The number of chunks original data divided into.
m =  The extra codes added to original data chunks to provide data protection.
n  =  The total number of chunks created after erasure coding process.
```

- 4+2 configuration ensures 66% usable capacity and allows for 2 OSD failures
- We are evaluating different configuration options (erasure coding vs replication) comparing performances, data durability and availability

Erasure coding su CEPH

Per realizzare erasure coding sui pool acceduti attraverso RBD (es. block device per VM), è necessario che un pool “tradizionale” di tipo replicato mantenga i metadati ed il journaling, mentre i dati vanno nel pool EC.

RBD can store image *data* in EC pools, but the image header and metadata still needs to go in a replicated pool.

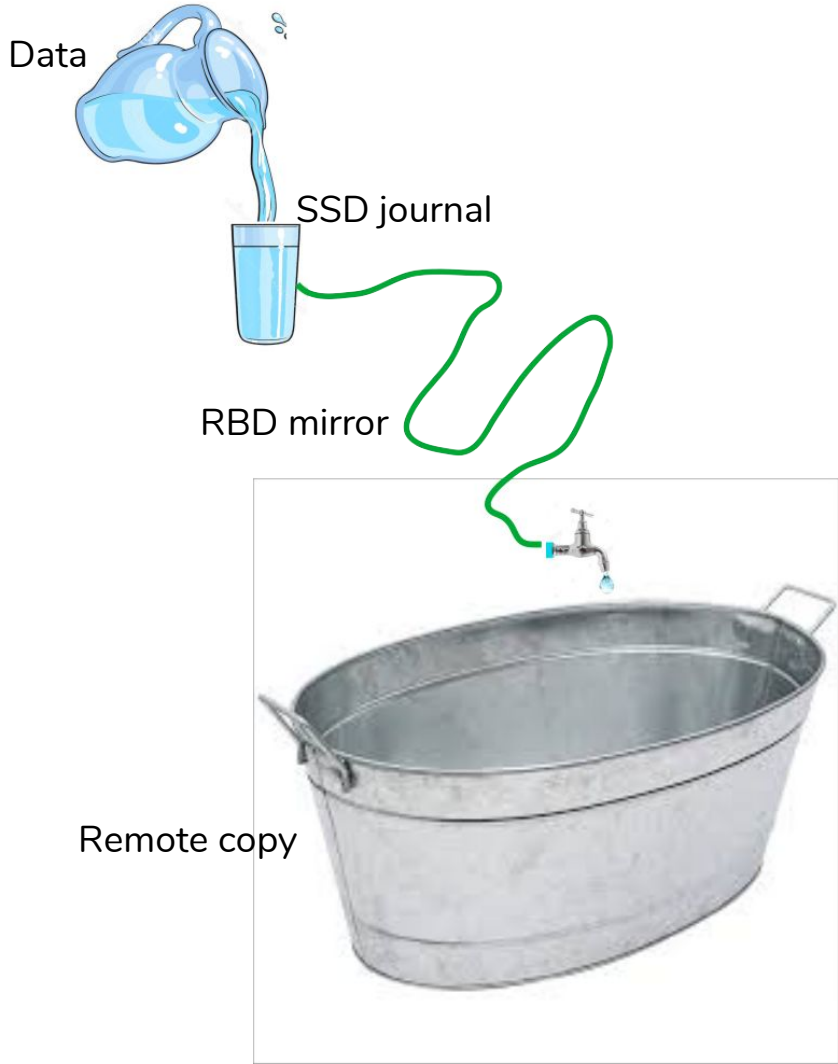
È necessario un numero piuttosto alto (6 nel caso descritto nella slide precedente) di storage node nel singolo data-center perchè la failure di un sistema di storage non impatti sull’accessibilità del dato

Erasure coding su CEPH

A prima vista, utilizzando i dischi SSD per il metadata pool le performance, almeno in scrittura, rispetto ad un pool replicato non sembrano differire sostanzialmente da quello di un pool tradizionale.

In corso misure per capire cosa succeda usando HDD per il metadata pool. Questo è importante perché per realizzare la replica remota di pool erasure coded non si possono usare dischi SSD per il metadata pool: in caso di scrittura intensa la cache SSD non fa in tempo a vuotarsi ed il pool SSD si riempie

- il mirror rbd è un processo asincrono e lento
- facile riempire il metadata pool se di dimensioni ridotte
- dopo un “upload” massiccio ci possono volere ore per la propagazione del journal sul pool remoto



Prestazioni dei vari tipi di volumi

È estremamente difficile fare valutazioni definitive sulle prestazioni dei diversi tipi di volumi, moltissimo dipende dallo use case. In generale possiamo dire:

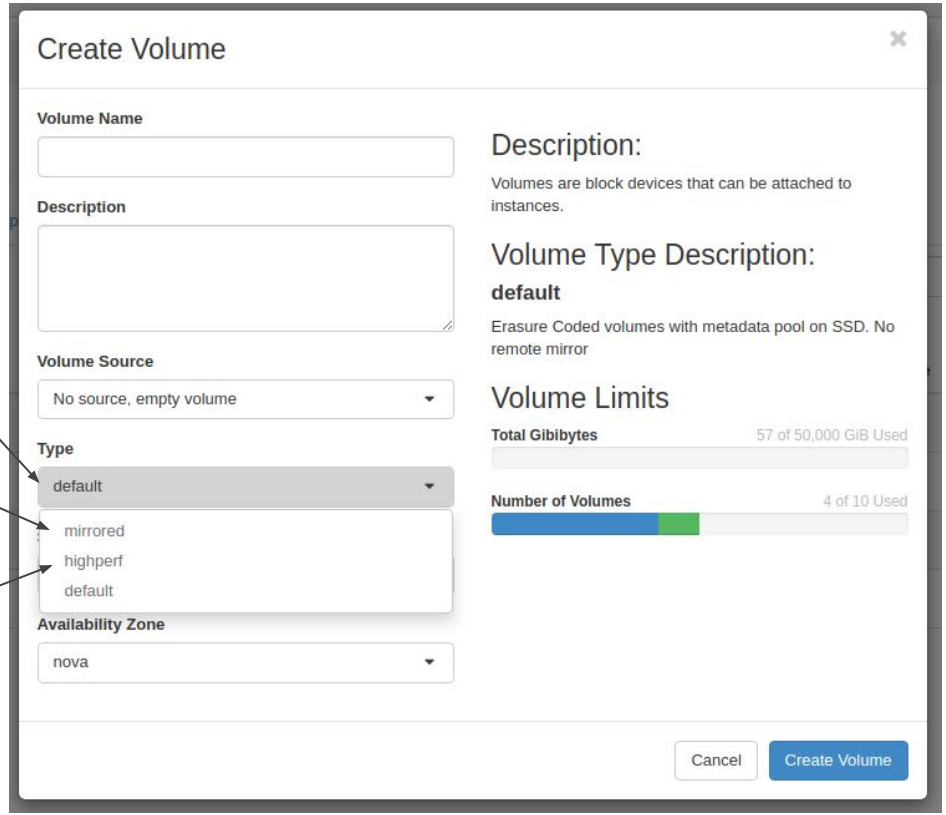
- l’accesso ai volumi cinder è più veloce rispetto all’accesso ai volumi nova
- la velocità di accesso in scrittura su pool erasure coded dipende dal backend usato sul metadata pool (SSD vs HDD)
- La velocità in accesso in lettura ai volumi replicati è migliore rispetto ai volumi erasure coded
- La velocità di accesso, soprattutto in lettura, dipende dal numero e dalle dimensioni dei file

Cinder Volume Types

default
Erasure Coded Pool, metadati su SSD

mirrored
Erasure Coded Pool, metadati su HDD, replica remota

highperf
Replicated (??) Pool, SSD



Future Work

Caratterizzare i setup descritti, ed altri, in termine di

- prestazioni
- affidabilità
- disponibilità
- resilienza

Decidere definitivamente quelli da adottare in produzione

Swift in INFN-CC

- **unico cluster** distribuito
- system metadata su disco SSD
- dati nativamente replicati nelle tre sedi
- accesso diretto allo storage ad oggetti attraverso API o client di livello più alto
- no accesso posix (eventual consistency)
- user metadata arbitrari

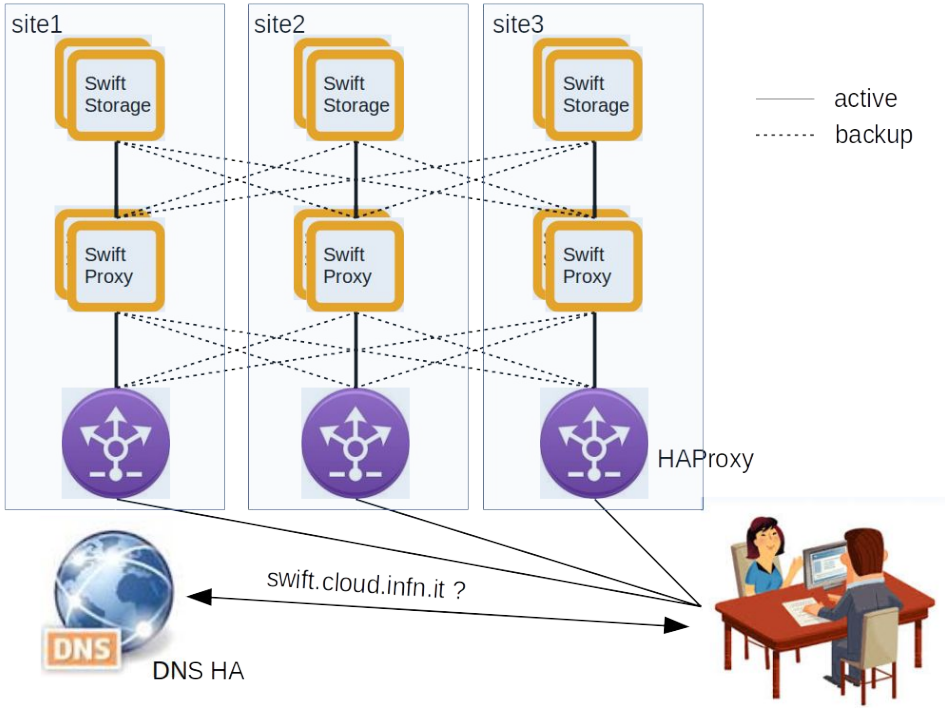
Sharing virtual appliances (and more..)

- **Swift Object Storage with 3 Regions**
  - 2 proxy nodes, 2 storage nodes (currently) per site.
  - RW affinity configured on the proxies in order to “prefer” the local storage servers
- Mixed SSDs and HDDs
  - *account* and *container* databases use SSD for better performance
- **Swift is used as backend for the Virtual Images repository in order to share the same images on all the sites**
  - VM snapshots are replicated as well
    - In case of a site failure, the remote snapshot replica can be used to start the VM on another site —> ***“cold”* migration**

Swift in INFN-CC

Use cases

- VM images per INFN-CC
- remote, encrypted backup
  - Duplicity
  - restic
  - CyberDuck
- long term data archiving
- ubiquitous data access



Reducing the storage costs

- Replica 3 is a typical configuration
  - ~33% of the raw capacity is usable!
- **Swift** **Erasure Coding** allows to achieve greater usable capacity

```
n = k + m  where ,
k  =  The number of chunks original data divided into.
m =  The extra codes added to original data chunks to provide data protection.
n  =  The total number of chunks created after erasure coding process.
```

- **8+4** configuration ensures 66% usable capacity and allows for **a whole site failure**
- We are evaluating different configuration options (erasure coding vs replication) comparing performances, data durability and availability

Erasure Coding su Swift

- perdita di performance non drammatica (non abbiamo numeri, ancora) rispetto alla normale replica
- i chunk sono distribuiti geograficamente
- soluzione ideale per backup e data archiving

Erasure Coding su Swift

- a seconda degli use case prevalenti, potrebbe diventare la modalità d'uso di default per Swift
- per ora, per creare un container erasure coded:

```
swift post -H 'X-Storage-Policy: Policy-2' my_data
```

Come Swift distribuisce i dati

Nel video qui sotto si capisce come Swift distribuisca i dati caricati, siano essi repliche di un file o chunk di file su container di tipo “erasure coded”. Mentre il dato è ridondante fin dal momento della creazione, la distribuzione definitiva delle repliche o dei chunk sui diversi nodi del cluster richiede tempi lunghi (> 1h).

https://gsbox.lngs.infn.it/s/RYQ969uX1KS6RYV

Conclusioni

- l’adozione della tecnologia erasure coding ci permette di **dimezzare il prezzo dello storage**
- l’obiettivo è di limitare al massimo (o, se possibile, eliminare) l’uso dei volumi/container replicati
- richiesta pianificazione accurata del setup
- performance ed affidabilità sono funzione del prezzo pagato per lo storage, ma non l‘unica variabile in gioco