Sperimentazioni su Cloud storage

… lavori in corso …

Giacinto DONVITO
INFN-Bari

Agenda

- Cosa si intende per cloud storage
  - Posix storage
  - Block Storage
  - Object Storage
- Overview delle varie soluzioni di cloud storage
  - GlusterFS
  - CEPH
  - Swift
- Test di funzionalità e di performance
- Considerazioni finali

Cosa si intende per cloud storage

- Per quanto riguarda lo storage, in una soluzione di cloud di tipo IaaS, si possono riconoscere almeno tre servizi che forniscono storage *persistente*:
  - Posix Storage:
  - Block Storage:
  - Object Storage
- C'è inoltre la possibilità di chiedere storage di tipo volatile:
  - Lo spazio disco dell'istanza di macchina virtuale running

Cosa si intende per cloud storage

- Posix Storage:
  - Permette la condivisione di file fra host diversi. In una IaaS viene usata di solito come back-end per vari servizi. Ad esempio la live-migration ha bisogno di un file-system posix per muovere le istanze running fra un host fisico e un'altro
  - Per piccole installazioni può essere suffciente un mount point NFS
  - Man mano che le installazioni scalano in dimensione, diventa subito necessario usare file-system paralleli che possano aumentare le dimensioni e le performance semplicemente aggiungendo hardware
  - Lo stesso file-system in alcuni contesti può essere montato dalle istanze virtuali running per permettere l'accesso a grandi quantità di dati
  - Nessuna soluzione di IaaS fornisce internamente una risposta a questa esigenza

Cosa si intende per cloud storage

- Block Storage:
  - Espone alle macchine virtuali un block device a basso livello.
  - Permette all'utente di decidere come formattare e usare il device.
  - Non può essere condiviso con più host contemporaneamente.
  - Il device deve essere disponibile sulla rete per essere raggiungibile da ognuno degli host virtuali
  - l'utente deve poter creare volumi on demand tramite la stessa interfaccia di gestione delle macchine virtuali
- Storage volatile:
  - Lo spazio disco dell'istanza di macchina virtuale running
  - Se non diversamente specificato, al riavvio dell'istanza tutte le modifiche saranno perse

Cosa si intende per cloud storage

- Object Storage:
  - Non permette l'accesso di tipo Posix (open, seek, write, close)
  - Non prevede la struttura a directory ma una organizzazione a bucket e oggetti
    - Bucket è un contenitore in cui scrivere/leggere gli oggetti
    - Non è possibile creare alberi di bucket (esiste un solo livello)
  - Gli oggetti sono file binari e possono essere descritti con metadati
  - I file possono essere acceduti attraverso web services (REST o SOAP)
    - Sfruttando il protocollo HTTP
  - Può essere usato come back-end per memorizzare le immagini delle macchine virtuali nei Market Place (o Image Service)
  - Può essere usato come sistema di basso livello per costruire applicazioni di più alto livello
    - Grazie anche alla presenza di APIs nei più diffusi linguaggi di programmazione
  - Solitamente non richiedono configurazioni con hw raid o simili
    - Gestiscono la ridondanza e le failures a livello software
  - È possibile fare upload di file di grandi dimensioni usando il Multipart Upload

Esempi di uso di Object Storage

- Creare un bucket/conteiner
  - curl -X PUT -i -H "X-Auth-Token: fc81aaa6-98a1-9ab0-94ba-aba9a89aa9ae" https://storage.swiftdrive.com/v1/CF_xer7_343/george
- Upload di un oggetto in un bucket/container:
  - curl -X PUT -i -H "X-Auth-Token: fc81aaa6-98a1-9ab0-94ba-aba9a89aa9ae" -T JingleRocky.jpg" https://storage.swiftdrive.com/v1/CF_xer7_343/george/JingleRocky.jpg
- Listare oggetti in un bucket/container:
  - curl -X GET -i -H "X-Auth-Token: fc81aaa6-98a1-9ab0-94ba-aba9a89aa9ae" https://storage.swiftdrive.com/v1/CF_xer7_343/george
    - HTTP/1.1 200 OK
    - X-Container-Object-Count: 3
    - X-Container-Read: .r:*,.rlistings
    - X-Container-Bytes-Used: 252732
    - Accept-Ranges: bytes
    - Content-Length: 53
    - Content-Type: text/plain; charset=utf-8
    - X-Trans-Id: txae17dfa78da64117aaf07585a1b02115
    - Date: Mon, 07 Nov 2011 23:00:56 GMT

    - JingleRocky.jpg
    - RockyAndBuster.jpg
    - SittingBuster.jpg

Cosa si intende per cloud storage

- Qualche esempio di tecnologie disponibili per ogni tipo di storage in esame:
  - Posix Storage
    - OpenStack: Nessuna soluzione di default
    - Amazon: Nessuna soluzione di default
    - Other: GlusterFS, Lustre, GPFS, CEPH
  - Block Storage:
    - OpenStack: Block Storage service (Cinder)
    - Amazon: Amazon Elastic Block Store (EBS)
    - Other: CEPH, iSCSI storage
  - Object Storage:
    - OpenStack: Object Storage service (Swift)
    - Amazon: S3
    - Other: CEPH, “GlusterFS UFO”

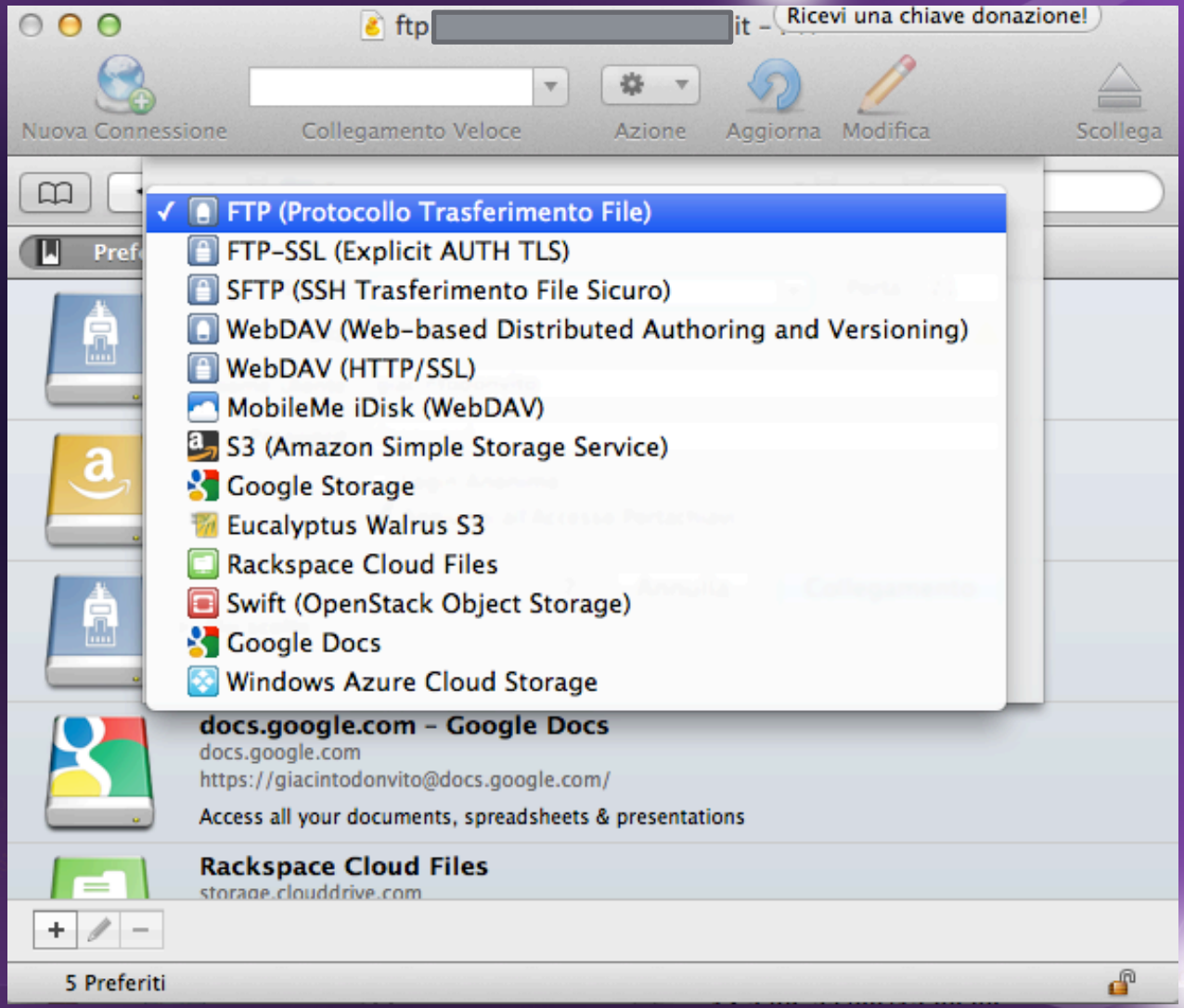
Storage Type

On-instance / ephemeral	Volumes block storage (Cinder)	Object Storage (Swift)
Used for running Operating System and scratch space	Used for adding additional persistent storage to a virtual machine (VM)	Used for storing virtual machine images and data
Persists until VM is terminated	Persists until deleted	Persists until deleted
Access associated with a VM	Access associated with a VM	Available from anywhere
Implemented as a filesystem underlying OpenStack Compute	Mounted via OpenStack Block-Storage controlled protocol (for example, iSCSI)	REST API
Administrator configures size setting, based on flavors	Sizings based on need	Easily scalable for future growth
Example: 10GB first disk, 30GB/core second disk	Example: 1TB "extra hard drive"	Example: 10s of TBs of dataset storage

il Cloud Storage secondo la declinazione di OpenStack

Nuovi protocolli di accesso

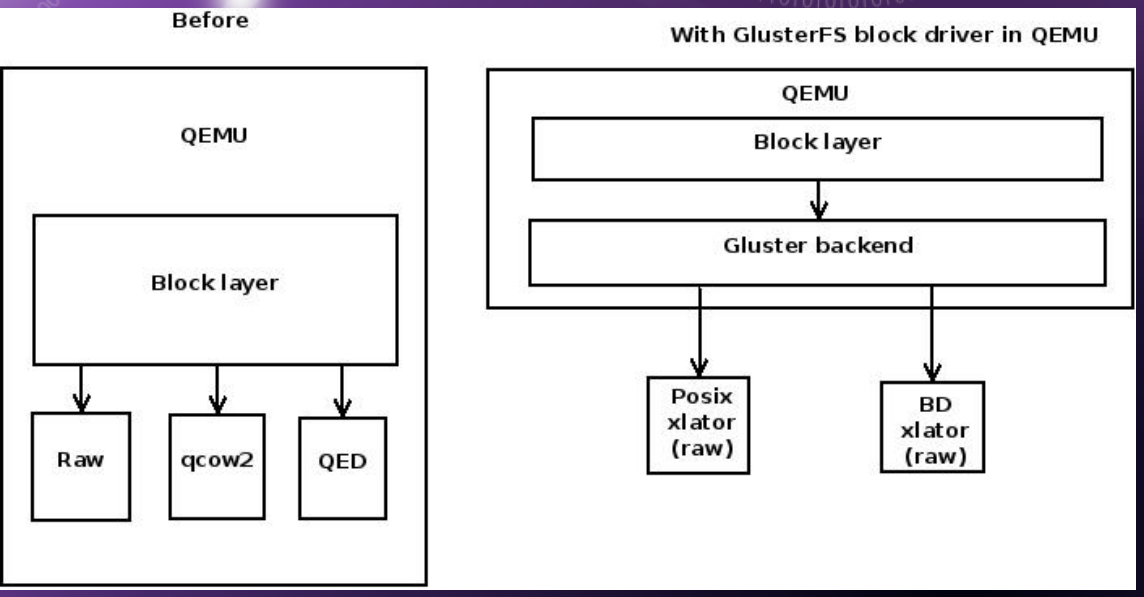
Cyberduck



GlusterFS

- Ufficialmente supportato (oltre che acquistato) da RedHat
- Molte soluzioni di IaaS lo supportano in modo più o meno ufficiale
  - Almeno come “storage posix”
  - In OpenStack c’è un driver che consente di usarlo come “Block Storage”
    - Integrato di Cinder da Grizzly in poi
- Funzionalità importanti per il contesto cloud:
  - Compatibilità posix per il running di macchine virtuali (per la live-migration)
  - Scalabilità a caldo
  - Supporto alla configurazione in replica software per resistere alle failures di interi server
    - Nessun “single point of failure”
  - Esiste un plugin che consente di usare GlusterFS per memorizzare dati usando l’interfaccia ad oggetti di Swift: Unified File and Object Storage (UFO)
    - In questa configurazione i dati scritti con l’interfaccia ad oggetti sono anche disponibili via posix
  - Possibilità di abilitare NFS come protocollo di export

GlusterFS



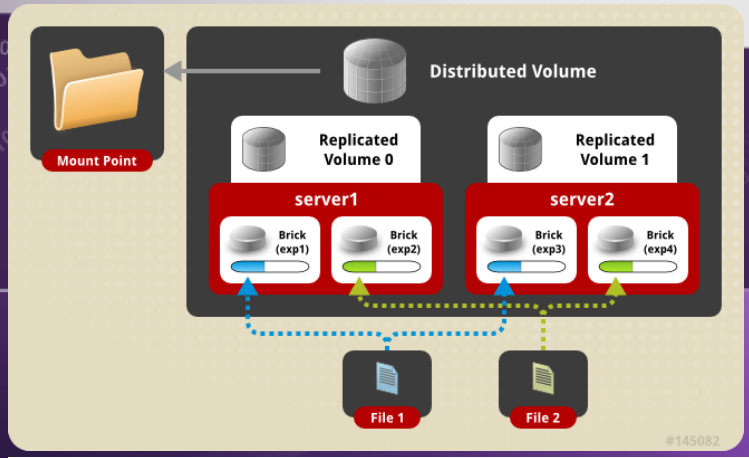


Figure 5.5. Illustration of a Distributed Replicated Volume

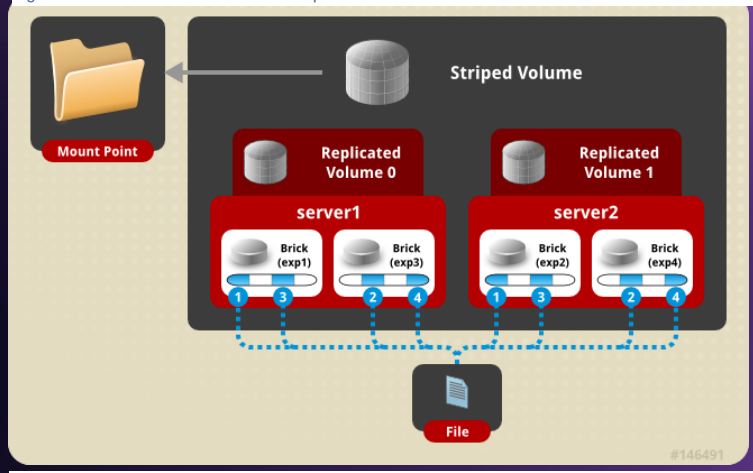


Figure 5.6. Illustration of a Striped Replicated Volume

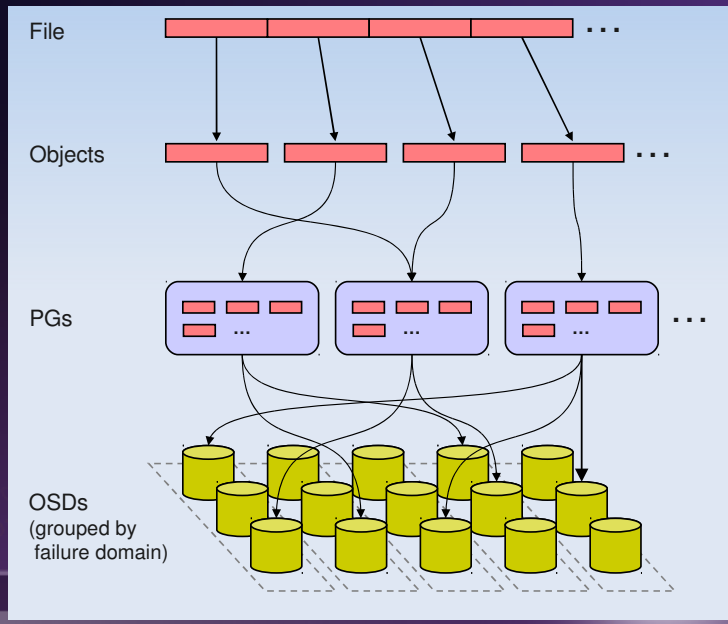
- Nella nuova versione (3.4) di GlusterFS ci sarà la possibilità di far interagire KVM/QEMU direttamente con GlusterFS senza passare per FUSE
  - Questo porterà ad un notevole miglioramento delle performance
- La possibilità di usare configurazioni (Distributed-Replicated o Striped-Replicated) consente di garantire ottime performance anche mantenendo una elevata resistenza alle failures

CEPH file-system

- Sviluppo cominciato nel 2009
- Attualmente acquistato da una company (Inktank), ma resta un progetto OpenSource e community driven
- È stato inserito di default nel kernel di Linux dalla 2.6.34 (maggio del 2010)
- Anche se ancora non consigliato in produzione, può usare come back-end *B-tree file system* (BTRFS)
  - Supporta molte funzionalità avanzate (Raid0/1 e a breve previsto 5/6, data deduplication…)
- Intrinsecamente disegnato per essere scalabile e fault-tolerant
  - È pensato per supportare più di 10'000 disk server!
  - Fino a 128 server di metadati! (stimate 250kops/s aggregate)
- La caratteristica peculiare di CEPH è quella di poter fornire tutti e tre i livelli di storage: Posix (sia a kernel level che con fuse), Block e Object storage
- Diverse soluzioni di cloud computing IaaS (p. es: OpenStack, CloudStack) supportano CEPH con driver appositi per sfruttare le funzionalità di Block Storage
- È pensato per non usare Raid hardware: la ridondanza è fatta a livello di software

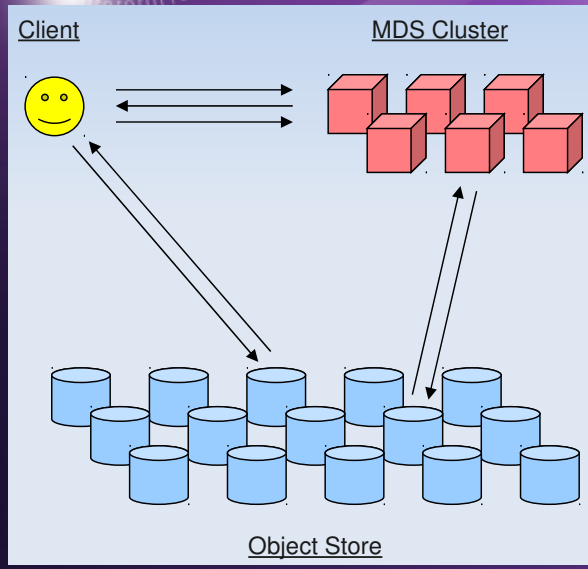
CEPH file-system

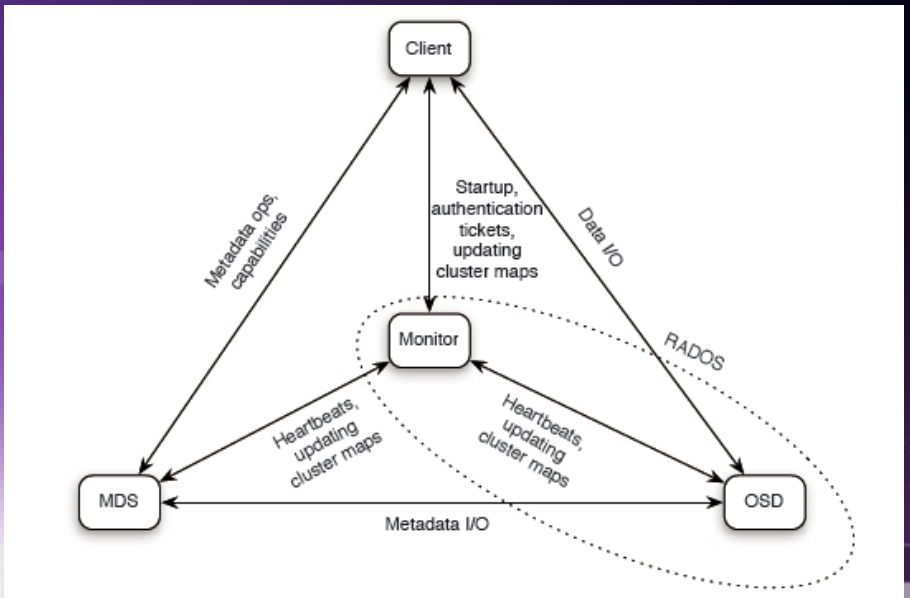
- In CEPH la distribuizione dei dati è fatta con una funzione hash
  - Quindi non è necessario fare query per sapere dove sono i dati
- Il mapping però è “instabile”:
  - Ogni aggiunta di disk server provoca un reshuffling
- È possibile configurare domini di fallimento su base: Disk, server, rack
- Il placement può essere organizzato secondo regole configurabili:
  - “le tre repliche devono essere posizionate in tre rack diversi”
  - Tutti i nodi del cluster sanno a quale server corrisponde un oggetto

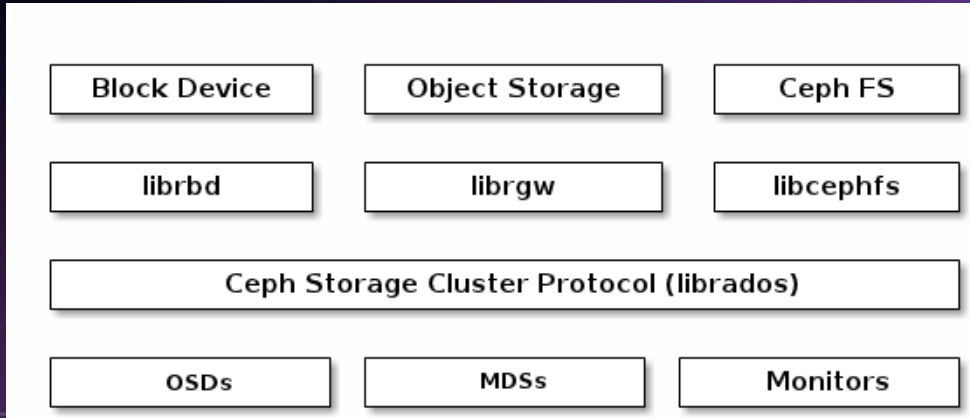


CEPH file-system

- Un classico cluster di 10 nodi potrebbe avere:
  - 3 monitor node
  - 3 MDS node
  - 8 server di storage
- Monitor: gestisce l'heartbeats fra i nodi
- MDS: gestisce l'I/O sui metadati
- OSD: contiene gli oggetti
- Il client interagisce con tutti e tre i servizi

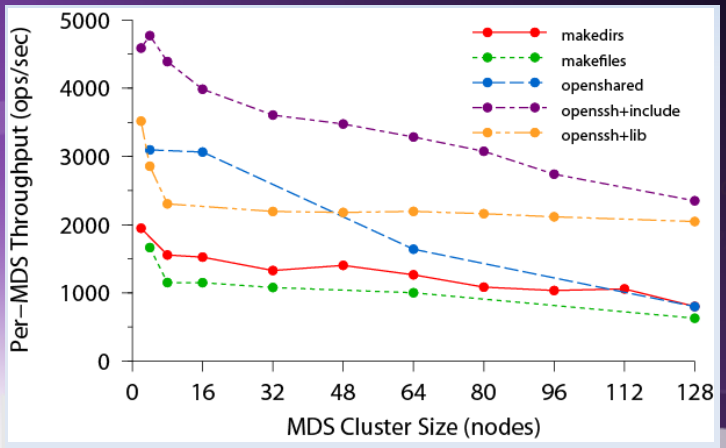


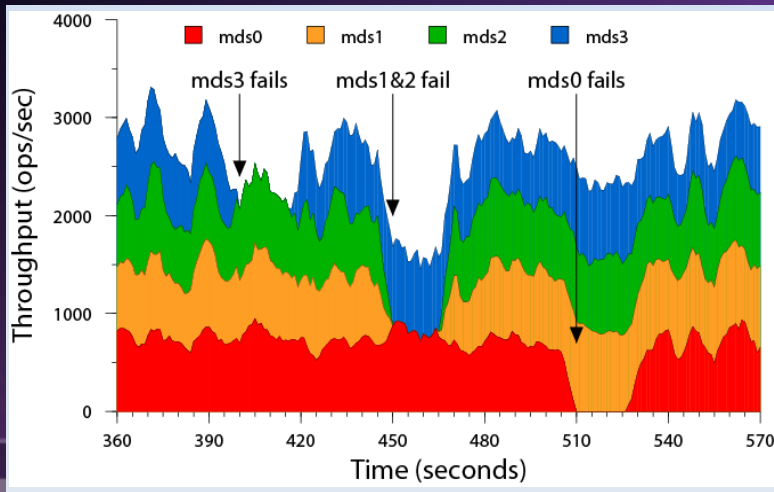




CEPH file-system

- Le tre interfacce sono semplici gateway di accesso che leggono e scrivono i dati usando le stesse primitive e la stessa libreria ad oggetti
- In CEPH gli oggetti si possono scrivere anche con una configurazione simile a RAID0 per migliorare le performance sul singolo file:
  - Object Size, Stripe Width, Stripe Count
- Data Scrubbing: periodicamente si possono controllare i dati su disco per assicurarsi che siano tutti in linea con i metadati e non ci siano state corruzioni
- Fully posix compliance => potrebbe supportare le home degli utenti o il running di macchine virtuali





CEPH file-system test

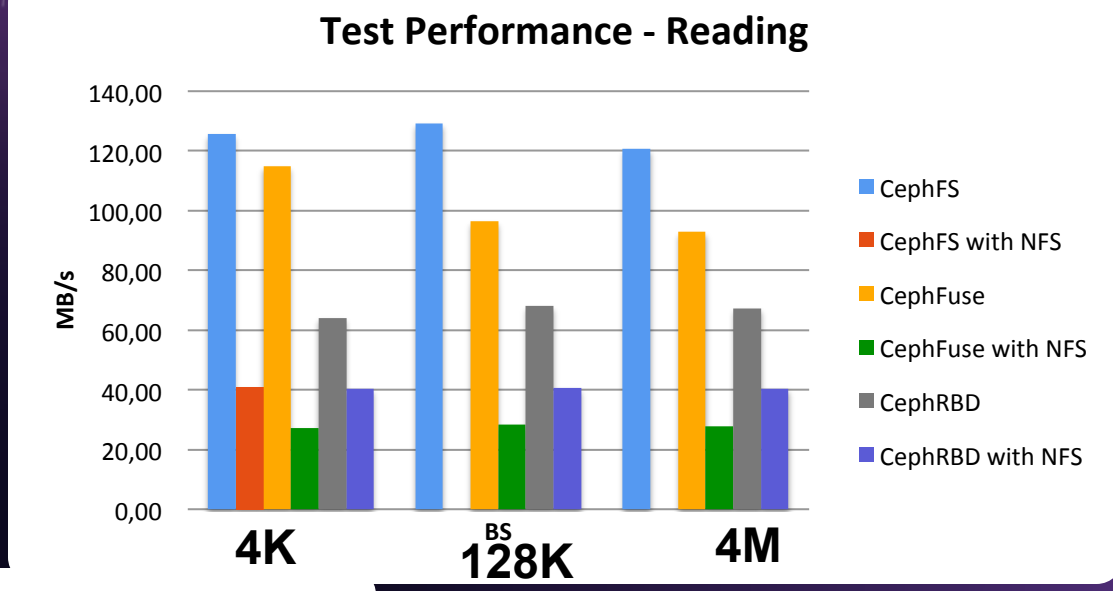
- Esiste il concetto di quorum: ogni servizio critico è previsto in un numero dispari di istanze.
  - Se 2 su 3 sono attivi il client può leggere e scrivere. Se solo uno è attivo, il client può solo leggere
- La verifica fatta a bari su un testbed di prova
  - l'alta affidabilità di questa soluzione ha funzionato perfettamente come previsto dalla documentazione
  - Provando sia failure del metadata server, sia dei data server che quelle dei monitor node
  - Sia con l'acesso posix che RBD
- Abbiamo anche verificato la possibilità di esportare in NFS il file-system
  - Funziona bene sia via RBD che posix usando FUSE
    - Usando il mount point kernel invece è molto instabile

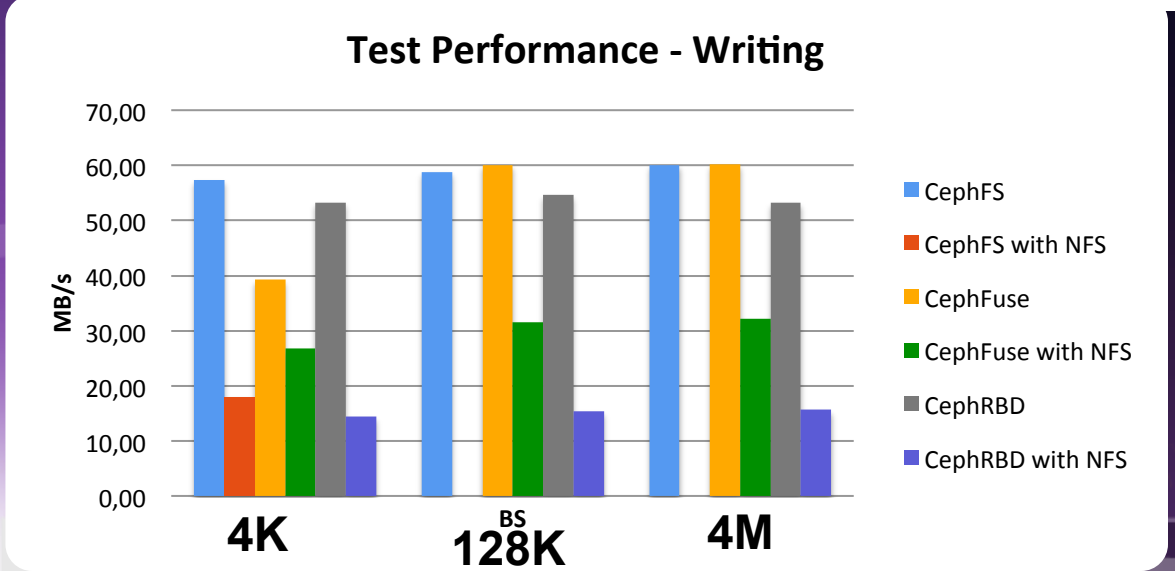
CEPH RBD

- Funzionalità di CEPH RBD:
  - Thinly provisioned
  - Resizable images
  - Image import/export
  - Image copy or rename
  - Read-only snapshots
  - Revert to snapshots
  - Ability to mount with Linux or QEMU KVM clients!
- In OpenStack CEPH può essere usato sia come device di Cinder (Block storage server) sia per le immagini virtuali di Glance (Image Service)

CEPH Performance test

BS
4K
128K
4M





CEPH Object Storage vs Amazon S3

Operazioni sui bucket

Features	Status	Note
PUT Bucket	Supportato	Crea un nuovo bucket.
DELETE Bucket	Supportato	Elimina un bucket.
GET Bucket	Supportato	Elenca gli oggetti presenti nel bucket.
GET Bucket ACL	Supportato	Elenca la ACL del bucket.
PUT Bucket ACL	Supportato	Imposta la ACL del bucket.
LIST Bucket Multipart Uploads	Supportato	Elenca le diverse parti di un oggetto in fase di uploading.
[GET|PUT|DELETE] Bucket CORS	Non supportato	Abilitazione e gestione dei CORS (Cross-Origin Resource Sharing)
[GET|PUT|DELETE] Bucket lifecycle	Non supportato	Gestione delle informazioni di configurazione di un bucket nel suo ciclo di vita.
[GET|PUT|DELETE] Bucket Policy	Non supportato	Creazione e gestione sugli accessi alle risorse dello storage.
[GET|PUT|DELETE] Bucket Tagging	Non supportato	Creazione e gestione di tag da legare al bucket per tracciare le sue attività.
[GET|PUT|DELETE] Bucket Website	Non supportato	Creazione e gestione di configurazioni web nel caso di utilizzo di un bucket come un host web.
[GET|PUT] Bucket Notification	Non supportato	Gestione delle configuazioni per le notifiche relative ad un bucket.
[GET|PUT] Bucket Logging	Non supportato	Gestione dei parametri di logging.
GET Bucket Object Version	Non supportato	Elenca metadati riguardanti tutte le versioni di oggetti presenti in un bucket.
PUT Bucket Versioning	Non supportato	Imposta lo stato della versione di un bucket esistente.
[GET|PUT] Bucket Request Payment	Non supportato	Operazioni che permettono ad un possessore di un bucket di far utilizzare la risorsa ad un richiedente.

Operazioni sugli Object

Features	Status	Note
PUT Object	Supportato	Aggiunge un oggetto al bucket.
Copy Object	Supportato	Crea la copia di un oggetto già presente nello storage.
DELETE Object	Supportato	Elimina un oggetto.
GET Object	Supportato	Ritorna un oggetto.
GET Object info	Supportato	Ritorna informazioni su un oggetto.
GET Object ACL	Supportato	Ritorna la ACL di un oggetto.
PUT Object ACL	Supportato	Aggiunge un oggetto al bucket
Initiate Multi-part Upload	Supportato	Inizializza una sessione di upload di un oggetto diviso in più parti.
Multipart Upload Part	Supportato	Operazione di upload di una parte di un oggetto.
Multipart Upload Part – Copy	Non supportato	Operazione di upload copiando i dati da un oggetto esistente.
List Multipart Upload Parts	Supportato	Da una lista delle parti di un oggetto che sono state caricate da un operazione di upload.
Complete Multipart Upload	Supportato	Completa una operazione di upload assemblando le parti caricate e quindi creando un nuovo oggetto.
Abort Multipart Upload	Supportato	Interrompe una operazione di upload di un oggetto diviso in più parti.

Swift Storage

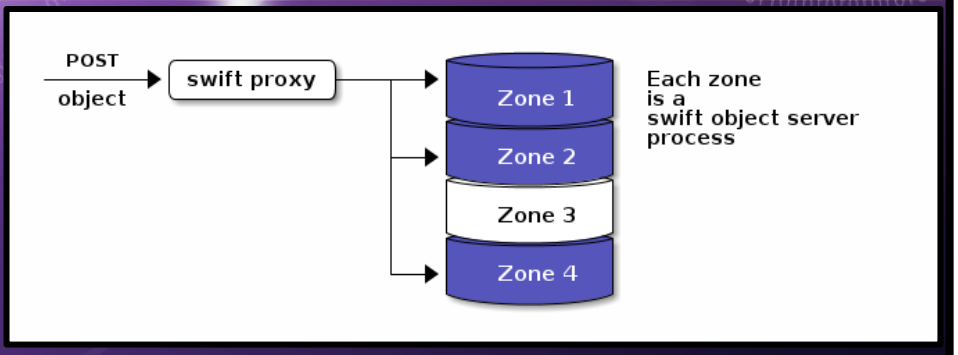
- Uno dei software OpenSource più usati e più completi per fornire un'interfaccia Object Storage
- Ne esistono installazioni "importanti": San Diego Computing Center
  - Più di 5 PetaByte e più di 50GB/s di bandwidth (dati del 2011!!)
- Funzionalità avanzate:
  - data availability basata sulla replica dei dati:
    - Possibilità di definire "zone" in modo da ottenere funzionalità di disaster recovery anche su WAN
  - Scalabilità sia dello spazio di storage che dei metadati
    - Il namespace non è in un DB o simili ma basato su "consistent hashing"
    - Possibilità di aggiunta di nodi di storage in modo trasparente
  - I nodi possono essere semplici file-server che scrivono binary files
  - REST API
  - Possibilità di accedere ad oggetti memorizzati nei "bucket"
  - API in molti linguaggi di programmazione (C/C#, php, python, etc)

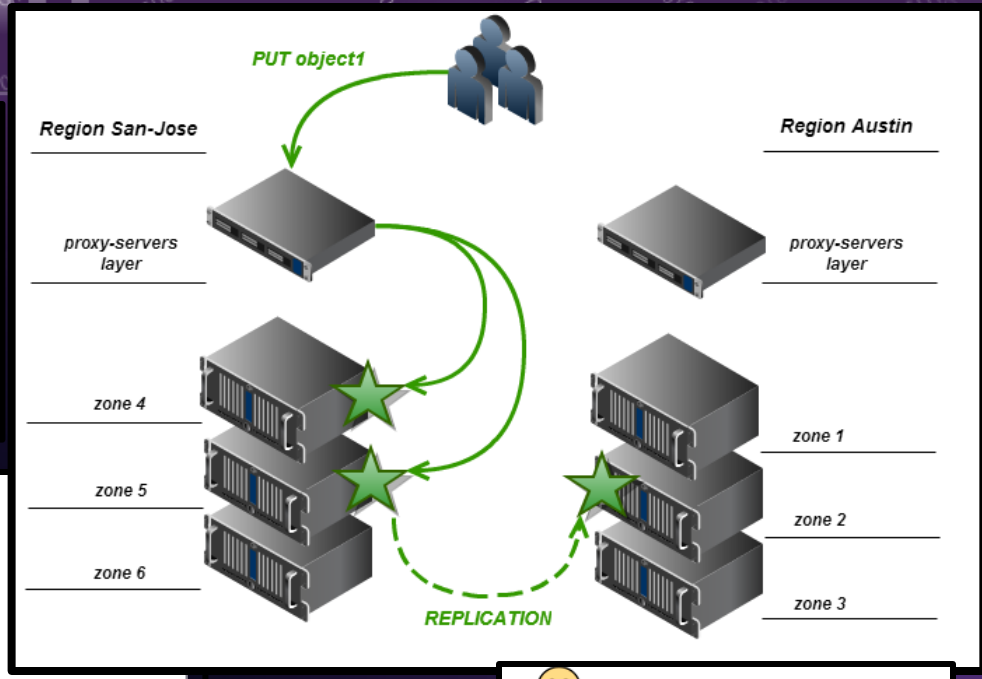
Swift Storage

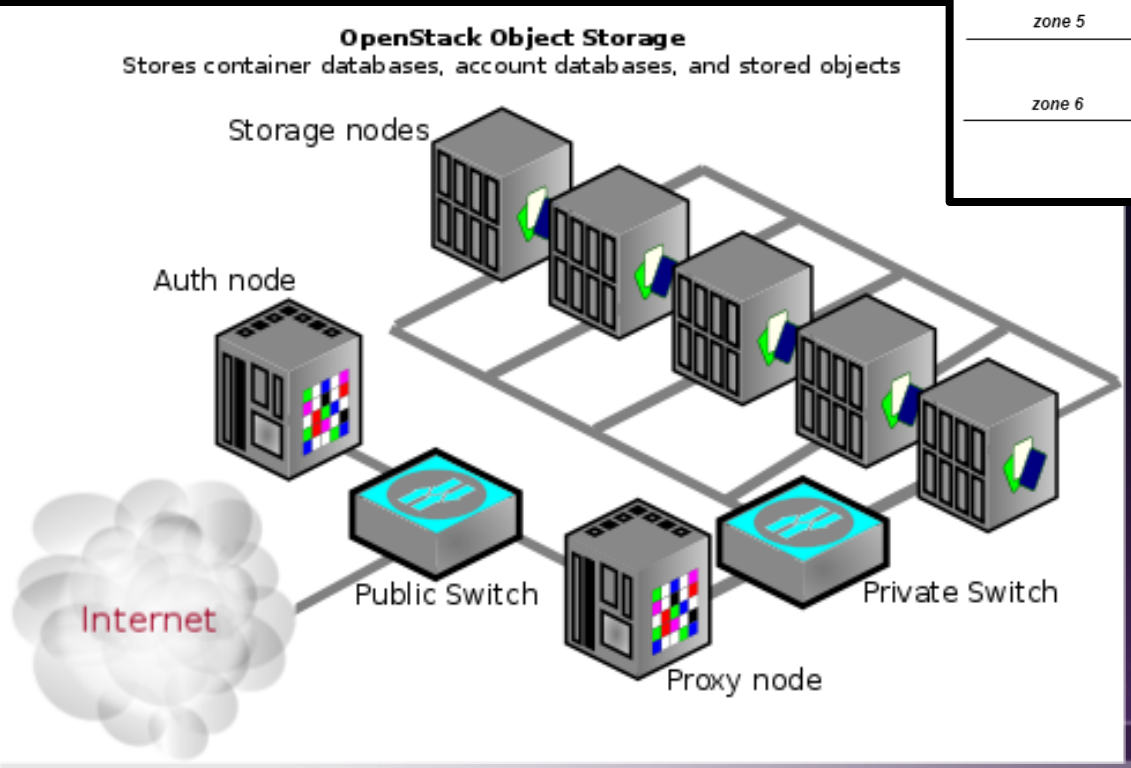
- Usa keystone per l'autenticazione
- Componenti di swift:
  - Proxy: espone le API all'esterno e forwarda le richieste allo storage che ha il dato richiesto
    - Possono esserci molti proxy server in un cluster
  - Account server: è il processo che mantiene l'informazione della lista degli account (ogni datanode ne ha uno)
  - Container server: è il processo che tiene l'elenco dei conteiner creati
  - Object server: è un BLOB storage server che può accedere, scrivere, cancellare gli oggetti
  - Rings: è una struttura dati che tiene traccia di dove si trovano i vari servizi (esiste uno specifico ring per ogni tipo di servizio)

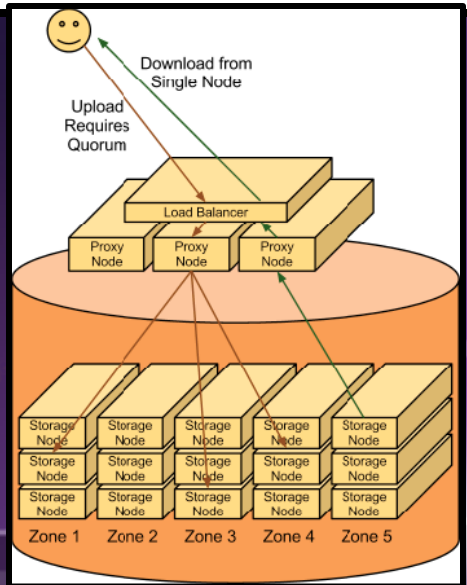
- GlusterFS usa un plugin per Swift per poter fornire una interfaccia Object Storage

Swift installation









Swift vs Amazon S3

- Usando Swift3 un modulo specifico di Swift è possibile emulare una gran parte delle API REST di Amazon S3 sullo Swift Object Storage. In particolare vengono supportate:
  - GET Service
  - DELETE Bucket
  - GET Bucket (List Objects)
  - PUT Bucket
  - DELETE Object
  - GET Object
  - HEAD Object
  - PUT Object
  - PUT Object (Copy)

SWIFT vs CEPH vs Gluster

- La stessa interfaccia di Swift si può installare su GlusterFS fornendo le stesse funzionalità togliendo la limitazione di 5GB di obj size
- Il data placement in GlusterFS è fatto con le regole di GlusterFS e non di Swift

	Swift	Ceph
Replication	Yes	Yes
Max. obj. size	5gb (bigger objects segmented)	Unlimited
Multi DC installation	Yes (replication on the container level only, but a blueprint proposed for full inter dc replication)	No (demands asynchronous eventual consistency replication, which Ceph does not yet support)
Integration /w Opentsack	Yes	Partial (lack of Keystone support)
Replicas management	No	Yes
Writing algorithm	Synchronous	Synchronous
Amazon S3 compatible API	Yes	Yes
Data placement method	Ring (static mapping structure)	CRUSH (algorithm)

CEPH vs SWIFT

- CEPH ha una interfaccia compatibile con le REST call di Swift

FEATURES SUPPORT

The following table describes the support status for current Swift functional features:

Feature	Status	Remarks
Authentication	Supported	
Get Account Metadata	Supported	No custom metadata
Swift ACLs	Supported	Supports a subset of Swift ACLs
List Containers	Supported	
Delete Container	Supported	
Create Container	Supported	
Get Container Metadata	Supported	
Update Container Metadata	Supported	
Delete Container Metadata	Supported	
List Objects	Supported	
Static Website	Not Supported	
Create Object	Supported	
Create Large Object	Supported	
Delete Object	Supported	
Get Object	Supported	
Copy Object	Supported	
Get Object Metadata	Supported	
Update Object Metadata	Supported	
Expiring Objects	Not Supported	
Object Versioning	Not Supported	
CORS	Not Supported	

REST vs POSIX

- http://www.enterprisestorageforum.com/storage-technology/file-system-interface-futures-how-the-cloud-impacts-our-world-1.html
- C'è qualcuno che si sta chiedendo se le APIs REST possano in qualche modo rappresentare l'interfaccia futura per l'accesso allo storage
- In campo scientifico, (non HEP) in effetti sta succedendo:
  - Molte applicazioni che usano workflow per analizzare dati stanno sfruttando con successo chiamate REST (webdav) per l'accesso allo storage
- In campo HEP??
  - In EMI una attività di sviluppo importante è stata orientata a permettere di usare webdav per accedere allo storage
  - Alcuni esperimenti fanno il download in locale del file per il processamento… questo modello sarebbe già compatibile con l'uso di HTTP+REST per accedere ai dati
- Cosa ci aspetta in futuro:
  - "We do not have a lot of POSIX file systems that scale today to 10s of PB and billions of files. There are three file systems in production with a parallel namespace (Gluster, PAN-FS, Lustre, and GPFS) and a new entry called Ceph."
  - Sembra che si possa immaginare un futuro in cui le REST call siano un importante sistema di accesso ai dati, e possa soppiantare POSIX in molte delle applicazioni che non hanno necessità di I/O parallelo

Conclusioni

- Nelle attuali infrastrutture di cloud computing non è sufficiente fornire una interfaccia allo storage di tipo posix
- Per le macchine virtuali l'interfaccia block è particolarmente importante
  - per le immagini dei sistemi operativi
  - per lo storage posix permanente
- L'interfaccia Object Storage si sta affermando come un'interfaccia di alto livello in grado di:
  - Permettere lo sviluppo di nuove applicazioni
  - Scalare sia in termini di numero di file, di performance e di spazio disco
  - Fornire funzionalità avanzate di gestione dello storage (multipart upload, ACL, metadati etc)

Credits

- Giacinto DONVITO (INFN-Bari)
- Giovanni MARZULLI (GARR/INFN-Bari)
- Stefano BUSSOLINO (Reply)
- Marco GAMBARINO (Reply)

PRISMA – PiattafoRme cloud Interoperabili per SMArt-government

PON04a2_A -PON04a2_A / F– Settore mart Cities and Communities and Social Innovation

Ministero dell'Istruzione dell'Università e Ricerca

PON Ricerca e Competitività 2007-2013

PRISMA

ReCaS

P.O.N. RICERCA E COMPETITIVA' 2007-2013
Azione I «Interventi di rafforzamento strutturale»
PONA3_00052, Avv. 254/Ric. – ReCaS

INFN

ISTITUTO NAZIONALE di FISICA NUCLEARE sezione di BARI