# Agenda

- Clustered Filesystems
- Block and Object: The Storage Evolution
- Introduzione a GlusterFS

# Sistemi di Storage: Clustered Filesystems

Un clustered file system è un file system che può essere collegato contemporaneamente a più server. Ci sono diversi metodi per creare un cluster di server ma la maggior parte di questi non prevede l'uso del cluster file system. Quando, però, il numero di nodi cresce e la complessità del cluster aumenta, il ricorso al clustered file system come risorsa condivisa può essere la soluzione più efficace.

# Che cos'è un Cluster?

Definizione  di Cluster:

Collezione di sistemi di calcolo indipendenti (workstations or PCs) collegati mediante una rete di interconnessione a basso costo (commodity interconnection Network), che viene utilizzata come una singola unificata risorsa di calcolo.

# Utilizzo dei Cluster

Possiamo far ricadere i cluster in tre famiglie che ne caratterizzano l’utilizzo:

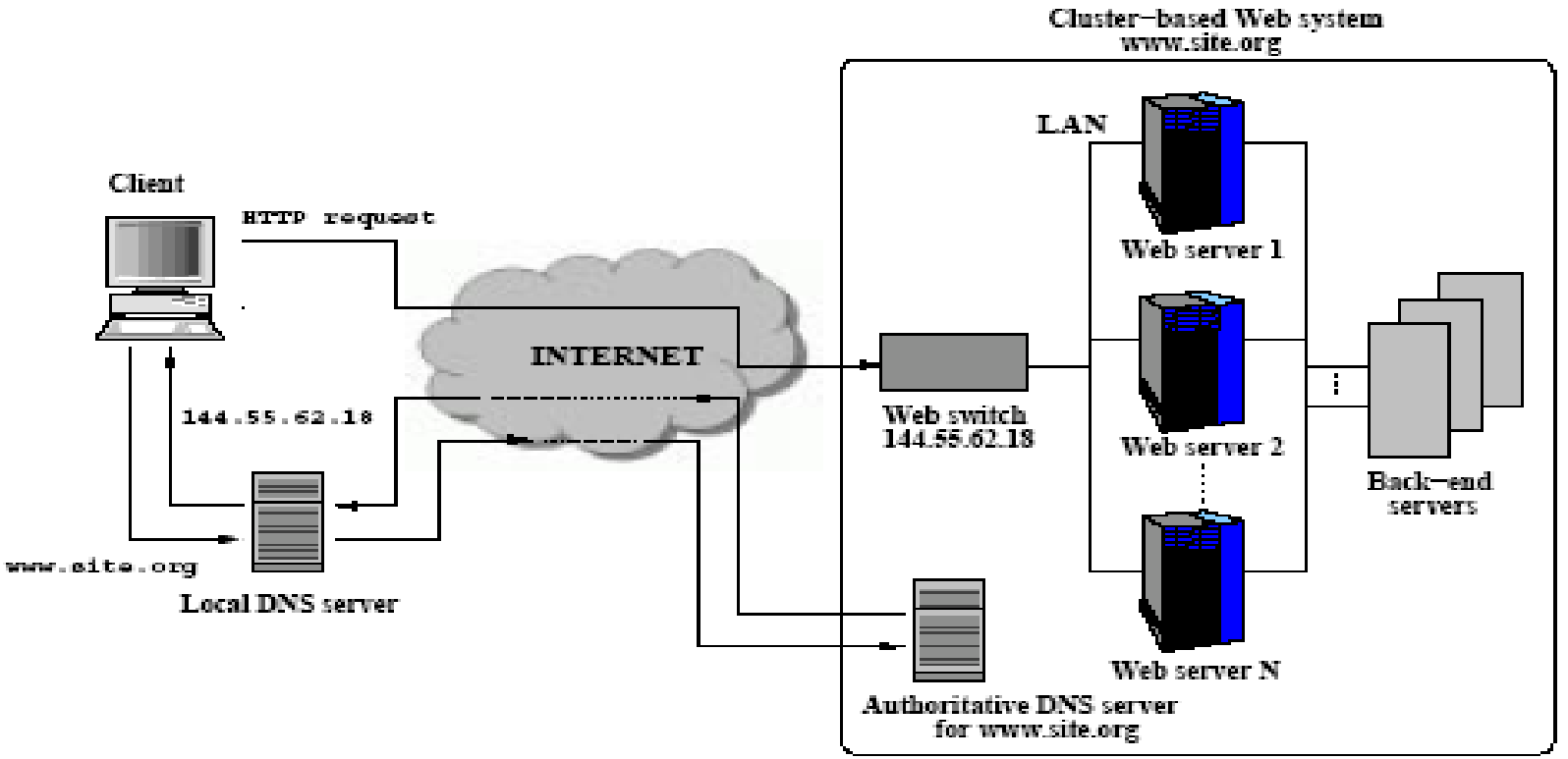
**High Availability Cluster**: i nodi offrono servizi ridondanti per garantirne la disponibilita'

**Load Balancing Cluster**: i nodi si spartiscono il carico di un determinato servizio dinamicamente

**High Performance Computing (HPC) Cluster**: i nodi eseguono in maniera coordinata programmi paralleli che fanno un uso intenso della CPU.

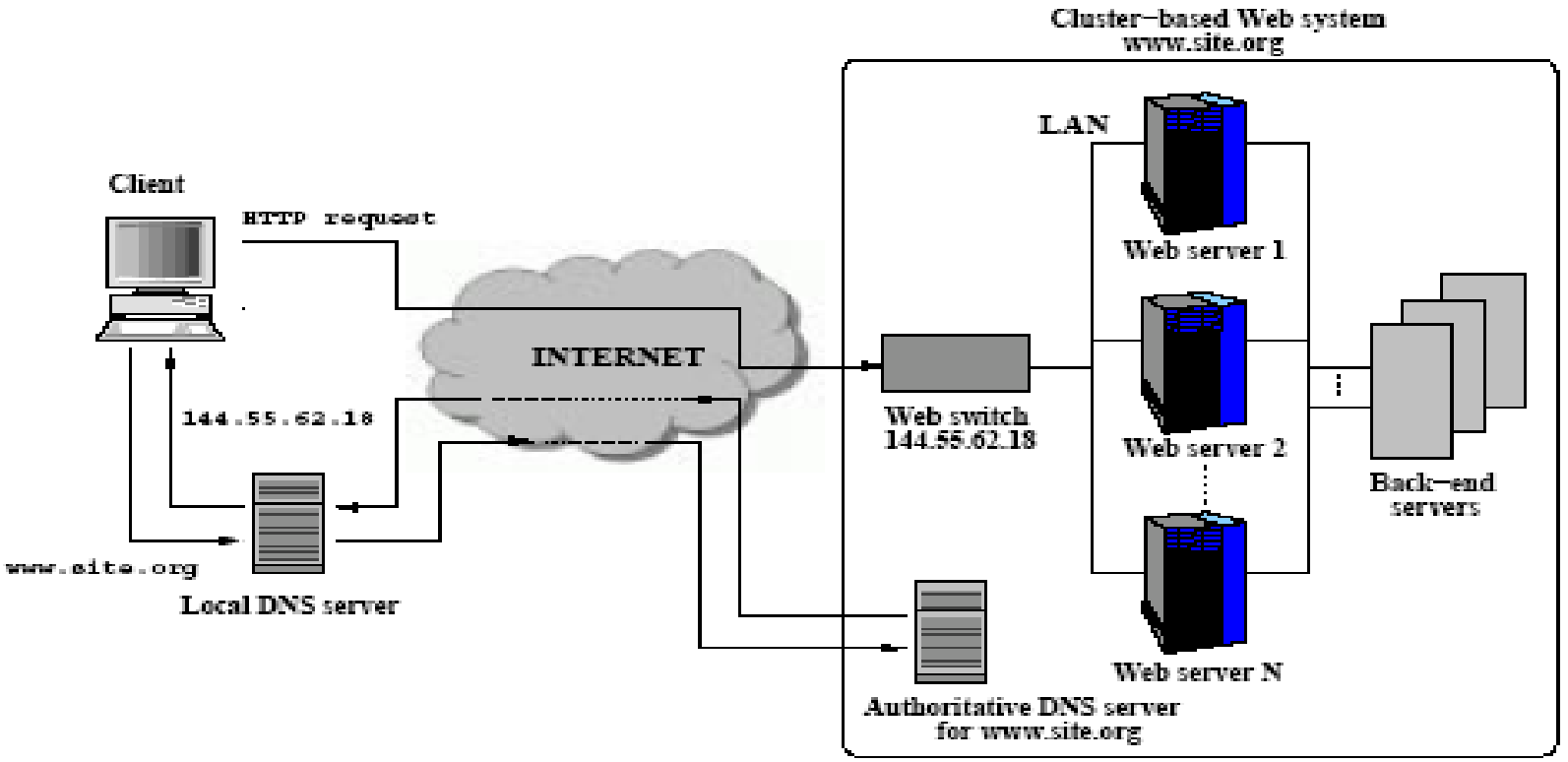
# Load Balancing Cluster

I nodi si spartiscono il carico di un determinato servizio dinamicamente



# High Availability Cluster

I nodi offrono servizi ridondanti per garantirne la disponibilita'



# Tipologie di Clustered Filesystems

**A disco condiviso**

La tipologia più utilizzata di cluster file system è a disco condiviso in cui due o più server accedono contemporaneamente ad un singolo sottosistema di storage che può essere un RAID o una SAN. Sono un esempio di questa tecnologia i file system VMFS e il Global File System.

**Senza alcuna condivisione**

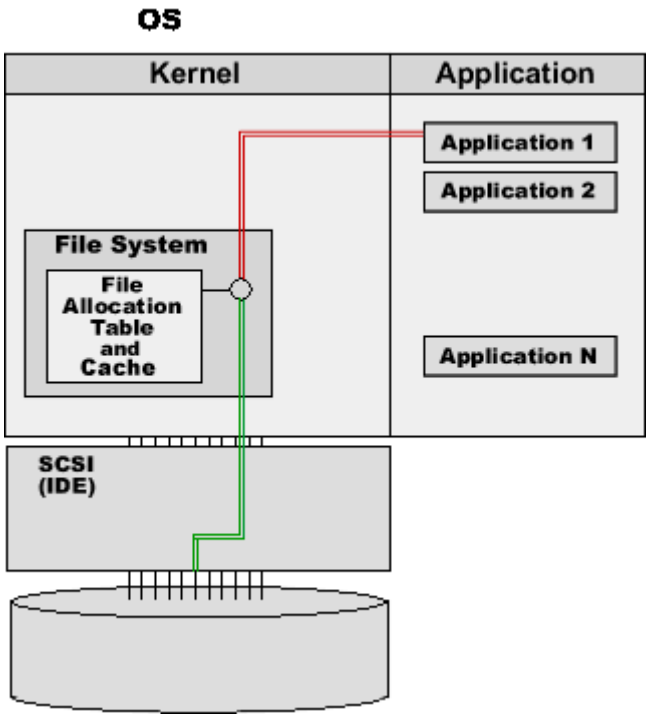
Un approccio completamente differente dal primo è quello di dotare ciascun sistema del proprio storage locale. Tutti questi storage sono poi sincronizzati fra loro attraverso la rete o un bus dedicato. Un file system globale è creato dalla cooperazione dei vari server come avviene, ad esempio, per i file system Isilon e IBRIX.

# Esempi di utilizzo

Gli scienziati che lavorano al **progetto ALICE del CERN di Ginevra** utilizzano una Storage Area Network con fabric Fibre Channel a 4 Gbit/s con un cluster file system per poter memorizzare la grandissima quantità di dati generati dall'esperimento (circa 1 GB/s per un mese). La scelta di questa architettura è stata dettata da un'alta garanzia di velocità, scalabilità e indipendenza dal vendor

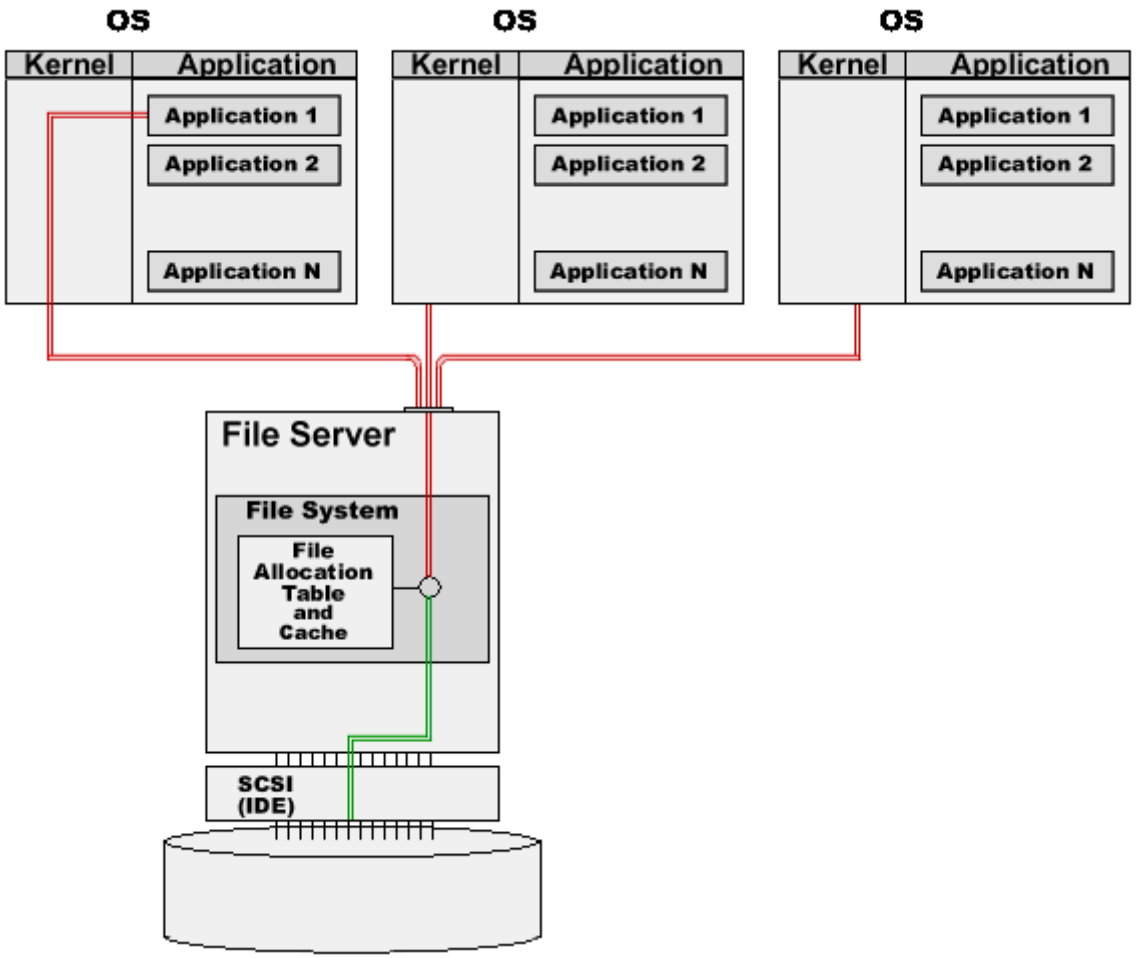
# Single OS Filesystems

Every modern Operating System (OS) has a component called a File System. That component is part of the OS kernel and it implements things like "files" and "file directories".There are many different File Systems, and they use various methods and algorithms, but the same basic functions are present in most File Systems



# NAS (Network Area Storage)

When server computers need to use the same data, a Network File System (also called NAS, or Network Attached Storage) can be used. The Network File System is implemented using a File Server and a network. The File Server is a regular computer or specialized OS that has a regular File System and regular disk devices controlled with this File System.



# Storage Area Network

Storage Area Network is a special type of network that connects computers and disk devices; in the same way as SCSI cables connect disk devices to one computer. Any computer connected to SAN can send disk commands to any disk device connected to the same SAN.

SAN provides Shared Disks, but SAN itself does not provide a Shared File System. If you have several computers that have access to a Shared Disk (via SAN or dual-channel SCSI), and try to use that disk with a regular File System, the disk logical structure will be damaged very quickly.

There are two main problems with Shared Disks and regular File Systems:

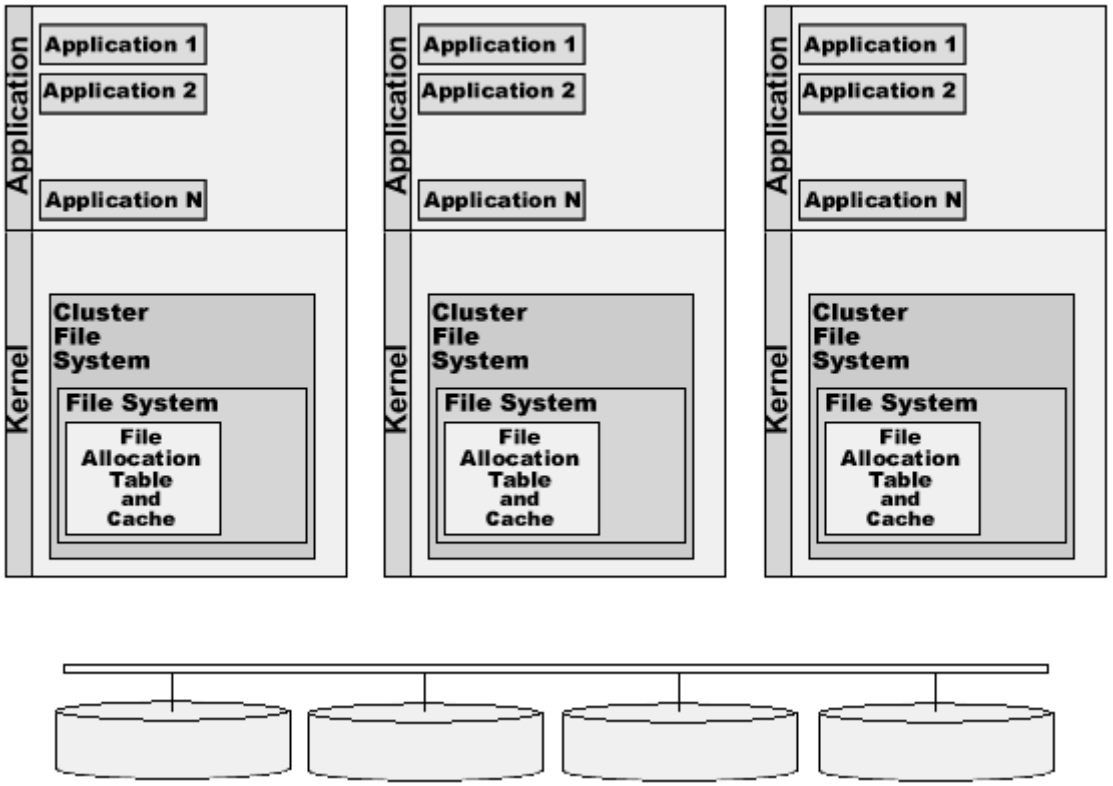
**Disk Space Allocation inconsistency**

**File Data inconsistency**

These problems make it impossible to use Shared Disks with regular File Systems as Shared File Systems. They can be used for **fail-over systems** or in any other configuration where only one computer is actually using the disk at any given time.

# Cluster File Systems

Cluster File Systems are software products designed to solve the problems outlined above. They allow you to build multi-computer systems with Shared Disks, solving the inconsistency problems.



# Cluster File Systems

The Cluster File Systems are usually implemented as "wrapper" around some regular File System. Cluster File Systems use some kind of inter-server network to talk to each other and to synchronize their activities. That inter-server "interconnect" can be implemented using regular Ethernet networks, using the same SAN that connects computers and disks, or using special fast, low-latency "cluster interconnect" devices.

The **Cluster File System solves the inconsistency problems and allows several computers to use Shared Disk(s) as Shared File System**.

Cluster File System products are available for several Operating Systems

# Esempi di Cluster File Systems

- GFS (Google Inc.)
- HDFS (Apache Software Foundation)
- Ceph (Inktank, Red Hat)
- MooseFS (Core Technology / Gemius)
- Windows Distributed File System (DFS) (Microsoft)
- FhGFS (Fraunhofer)
- GlusterFS (Red Hat)
- Lustre
- Ibrix

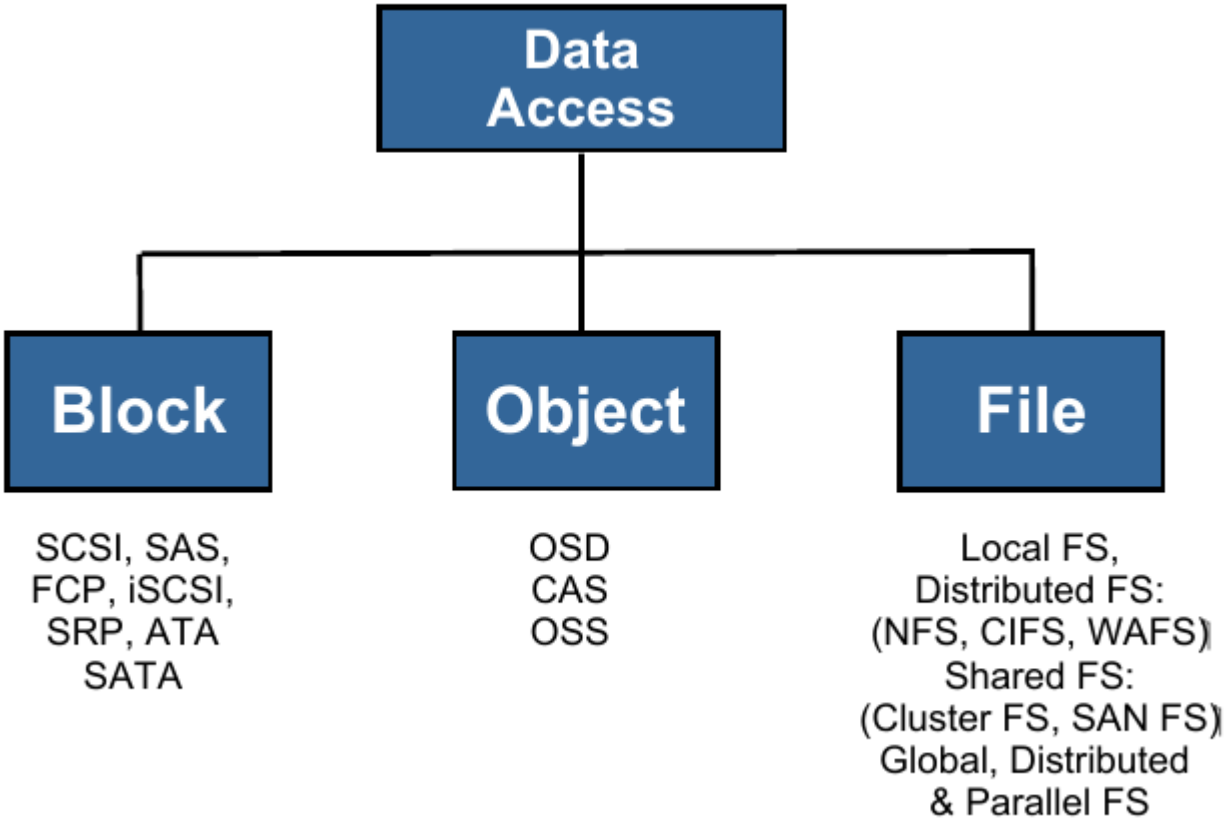
# The Storage Evolution

- Block Storage
- Files Storage
- Object Storage

Ovvero

- Block-Based Data Access
- File-Based Data Access
- Object-Based Data Access

# Data Access Taxonomy



# Block Storage

Block Storage is persistant storage organized into **unstructured "blocks", each the same length**. An ordinary disk drive, RAID array, or USB storage key are examples of locally attached "block storage".

Block storage can be either "locally attached", or it can be "network" attached, in a SAN, speaking a network protocols such as iSCSI.

Block storage devices typically are formatted with a filesystem, such as Linux's ext3 or btrfs, or Microsoft's FAT32 or NTFS. The Linux filesystems such as ext3 implement the POSIX filesystem semantics.

# Block Storage

In **OpenStack**, block storage is provided by the Nova system working with the **Cinder** system. When you start a Nova compute instance, it will probably come with some block storage devices by default, at the very least to hold the read/write partitions of the running OS.

These block storage instances can be "ephemeral" (the data goes away when the compute instance stops) or "persistant" (the data is kept, can be used later again after the compute instances stops), depending on the configuration of the OpenStack system you are using.

A block storage device usually **can be attached and in read/write use by only one machine or compute instance at a time**.

# Object Storage

Object Storage is persistant storage of "objects" in a way that is useful for HTTP access and for making guarantees about the safe storage of data. **An object is a stream of bytes, with an associated name, a MIME type, an access control list (ACL), and other HTTP-related and random metadata**. Once an object is created and written, it cannot be changed, only copied or deleted.

Object Store systems **are often configured to make very strong guarantees that the data will not be lost**, even in a disaster. Keeping three copies across two geographically separated datacenters is common.

# Object Storage

**Once written, an object can be read by many clients at once**. If you want to "fan out" data: writing it once, and having it read many times by many machines in the near and far future, object storage can help implement that pattern.

With some care in the naming of an object, and by pointing the DNS CNAME of the domain name part of a URL at the address of an Object Storage service, the object can be retrieved and displayed by any web browser on the internet. A great deal of the content, especially graphics and icons, that display in your web browser, are actually objects being served out of Amazon's AWS S3 service.

In **OpenStack**, the object storage system is one of the two original projects, and is named **Swift**.

# Object-Based Data Access

- Object-Based Storage Devices (OSD)
- Object Storage Systems
  - Object Storage Server (OSS)
  - Content Addressable Storage (CAS)
  - Content Aware Storage (CAS)

# Clustered File System vs. Distributed File System

A **clustered file system** is a file system which is shared by being simultaneously mounted on multiple servers.

A **distributed file system** or network file system is any file system that allows access to files from multiple hosts sharing via a computer network.

This definition is confusing. Is distributed file system a single large file system that spans over multiple hosts and presents a unified view (global namespace)? How exactly are these semantics implemented. Examples will be helpful.

# Clustered File System vs. Distributed File System

They both provide a unified view, global namespace, whatever you want to call it.

**The difference** lies in the model used for the underlying block storage. **In a cluster filesystem** such as GFS2, **all of the nodes connect to the *same* block storage**, with access mediated by locks or other synchronization primitives.

**In a distributed filesystem** such as GlusterFS**, each server has its own *private* block storage, which is only unified at a higher level**.

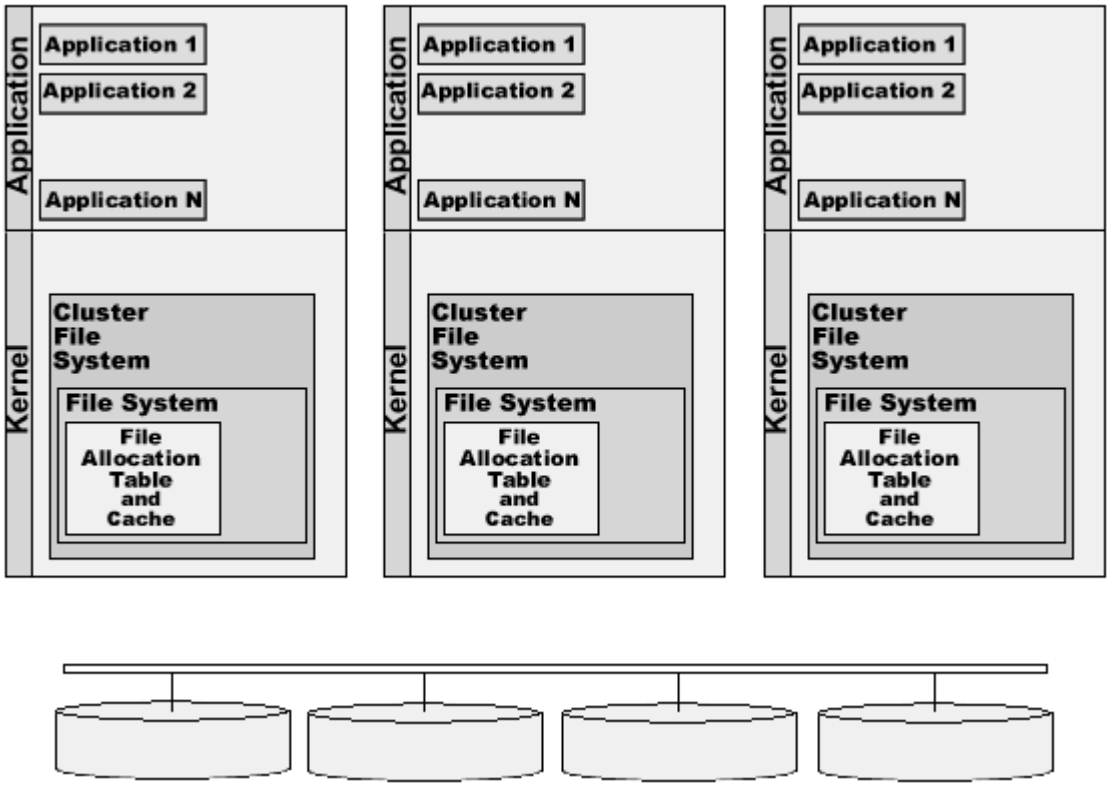
# Distributed File Systems

Distributed file systems **do not share block level access to the same storage but use a network protocol**.

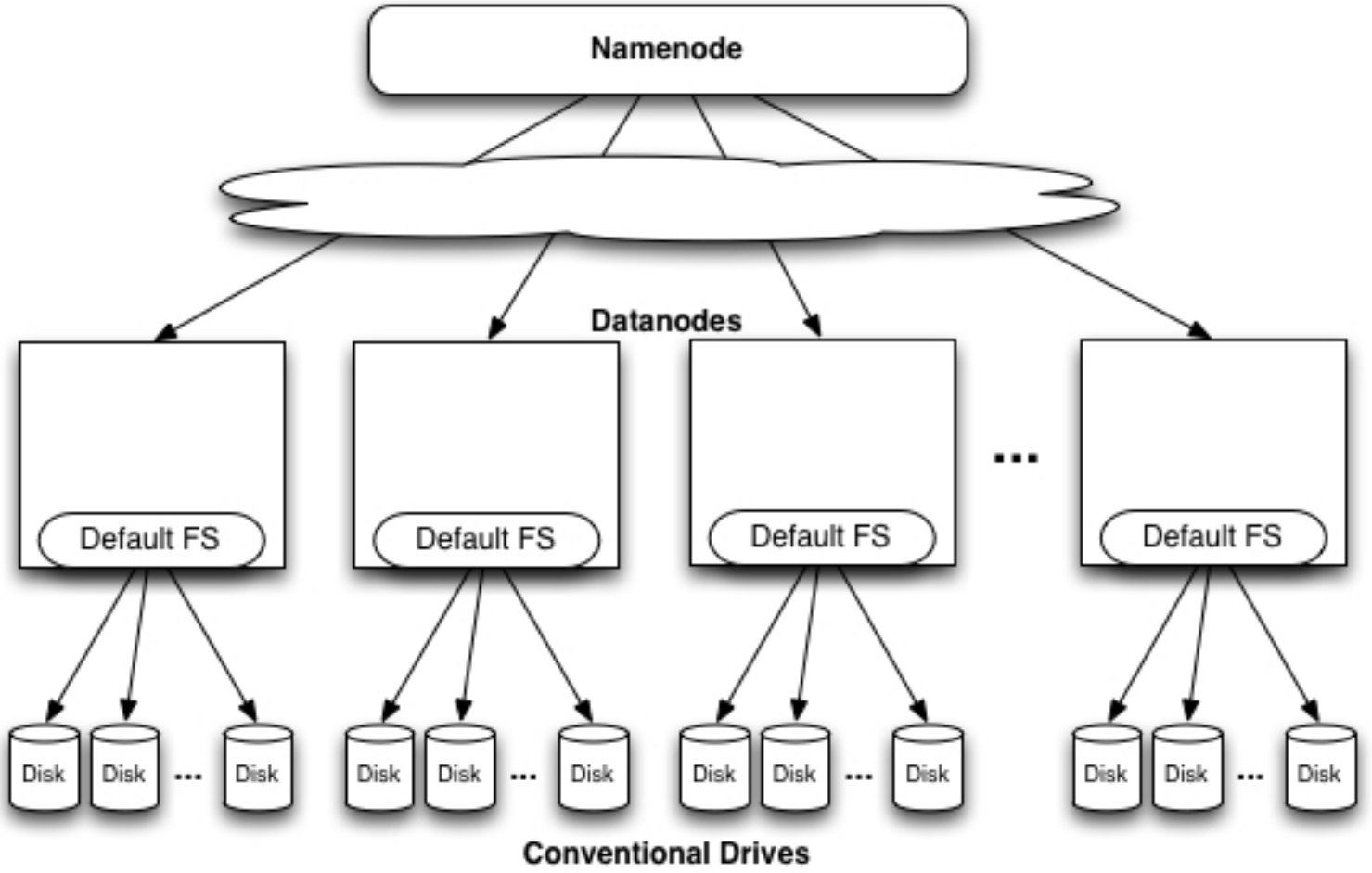
These are commonly known as **network file systems**, even though they are not the only file systems that use the network to send data.

Distributed file systems can restrict access to the file system depending on access lists or capabilities on both the servers and the clients, depending on how the protocol is designed.

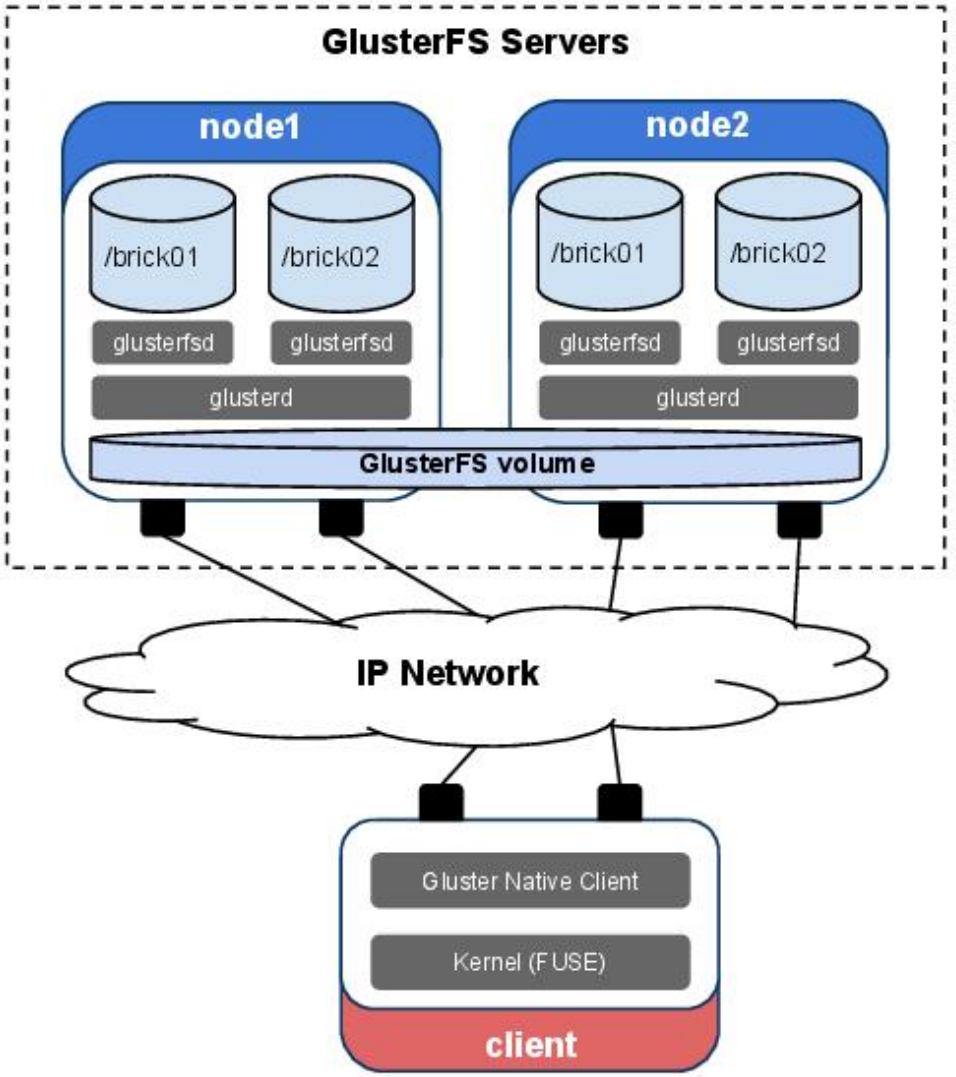
# Clustered Filesytem



# Distributed Filesytem (Google Hadoop)



# Distributed Filesystem (GlustrFS)



# Clustered File System vs. Distributed File System

**Cluster filesystems** have mostly fallen out of fashion, primarily because their storage model requires a relatively expensive external (e.g. FC/iSCSI) disk subsystem plus switches, adapters, etc. The up side is that this allows disk failures to be handled on the external subsystem, and the same-ness of the underlying storage can ease handling of server failures as well.

**Distributed filesystems**, on the other hand, can be and usually are built using cheaper SATA/SAS disks through on-board controllers. While such filesystems can easily beat their cluster cousins in terms of throughput per dollar, they often do so at the cost of worse latency and greater complexity to provide data availability across separate pools of storage.

# La scelta dello Storage

Come scegliere **lo storage in un virtual data center**?

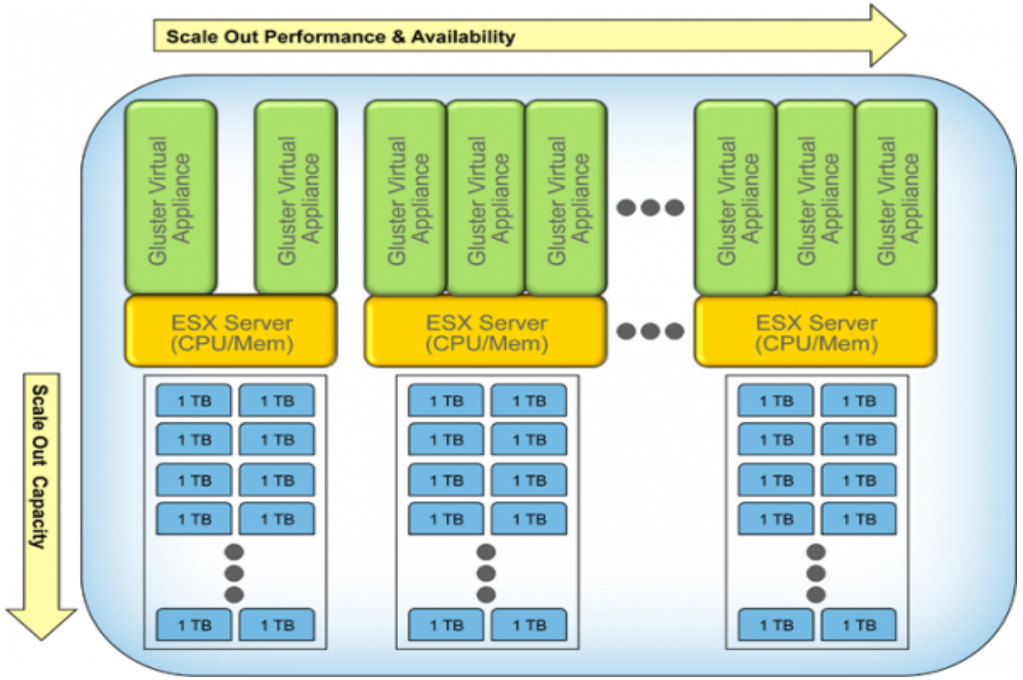
Prediligendo una soluzione orientata alla **High Availability**

Es. Per il servizio di provisioning di VM, è molto utile configurare l'infrastruttura per permettere la live migration di VM tra compute node.

Si definisce come "**live migration**" l'operazione attraverso la quale è possibile migrare una macchina virtuale dal server fisico sul quale è in esecuzione (sorgente) verso un server fisico differente (destinazione). Per permettere questa operazione, è necessario che l'area dati che ospita le macchine virtuali sia condivisa tra il server sorgente e il server destinazione. Questa area dati è quindi caratterizzata da alto I/O e dal fatto che deve essere condivisa tra più server.

# La nostra scelta: GlusterFS

GlusterFS è un file system open source distribuito e scalabile orizzontalmente, la cui capacità può essere dinamicamente espansa mediante l'aggiunta di nuovi nodi. GlusterFS è in grado di arrivare a gestire fino a diversi peta byte, migliaia di client e diverse aree dati (storage) organizzandole in blocchi che rende accessibili su Infiniband RDMA (remote direct memory access, fibra ottica) o connessioni TCP/IP.




**Aumentando i nodi si aumentano performance e availability.**

**Aumentando i dischi per nodo si aumenta la capacità complessiva dello storage.**

# Perchè GlusterFS

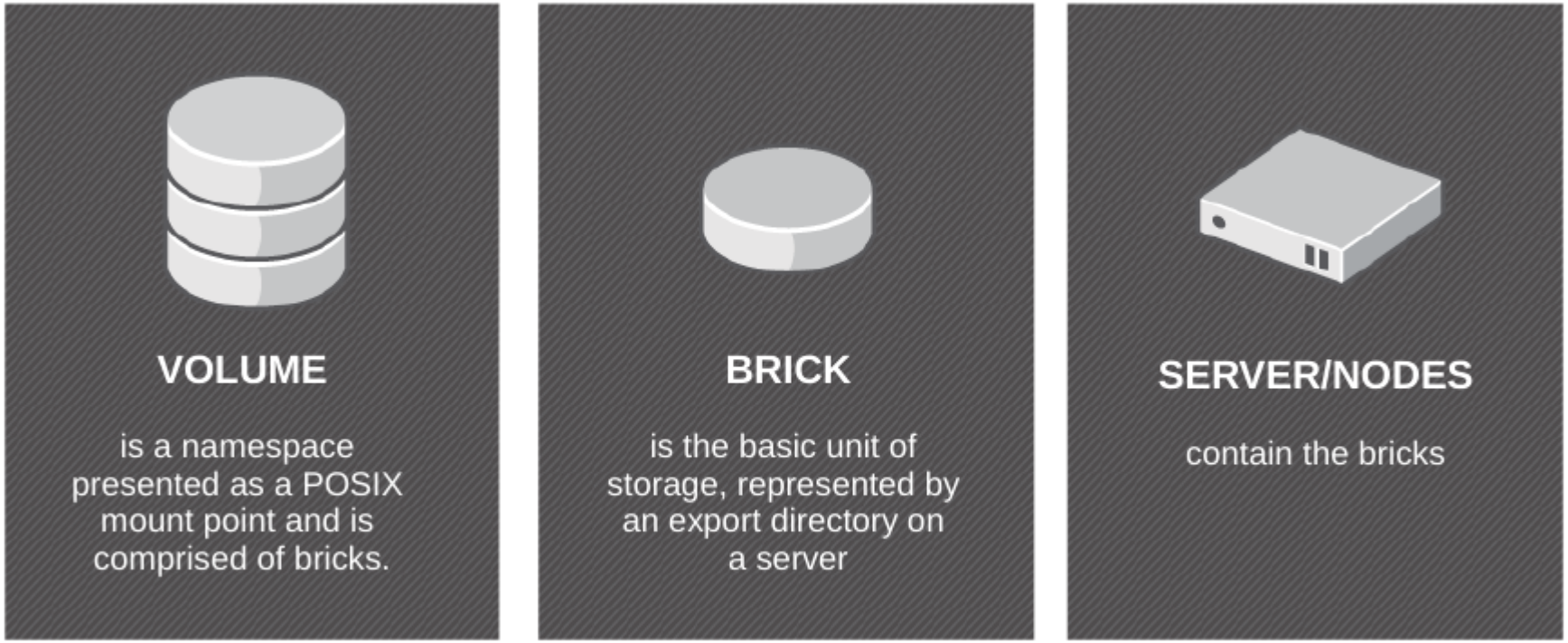
**Scalable –** Absence of a metadata server provides a faster file system.

**Affordable –** It deploys on commodity hardware.

**Flexible –** As I said earlier, GlusterFS is a software only file system. Here data is stored on native file systems like ext4, xfs etc.

**Open Source –** Currently GlusterFS is maintained by Red Hat Inc, a billion dollar open source company, as part of Red Hat Storage.

# GlusterFS Concepts




Nel file system Gluster, il termine con cui si identifica una risorsa condivisa è **volume**, ossia un insieme logico di blocchi (**bricks**), dove per blocco si intende una directory esportata da un **server** compreso nel pool degli storage fidati (trusted storage pool).

# Storage concepts in GlusterFS

- **Block Storage** – They are devices through which the data is being moved across systems in the form of blocks.
- **Cluster** – In Red Hat Storage, both cluster and trusted storage pool convey the same meaning of collaboration of storage servers based on a defined protocol.
- **Distributed File System** – A file system in which data is spread over different nodes where users can access the file without knowing the actual location of the file. User doesn't experience the feel of remote access.
- **FUSE** – It is a loadable kernel module which allows users to create file systems above kernel without involving any of the kernel code.
- **POSIX** – Portable Operating System Interface (POSIX) is the family of standards defined by the IEEE as a solution to the compatibility between Unix-variants in the form of an Application Programmable Interface (API).
- **RAID** – Redundant Array of Independent Disks (RAID) is a technology that gives increased storage reliability through redundancy.

# Storage concepts in GlusterFS

- **server**: la macchina (virtuale o reale) che ospita il filesystem ed all'interno della quale verranno registrati i dati;
- **client**: la macchina che monta il volume (che può agire anche da server);
- **glusterd** – glusterd is the GlusterFS management daemon which is the backbone of file system which will be running throughout the whole time whenever the servers are in active state.
- **Brick** – Brick is basically any directory that is meant to be shared among the trusted storage pool.
- **Trusted Storage Pool** – is a collection of these shared files/directories, which are based on the designed protocol.
- **Volume** – A volumes is a logical collection of bricks. All the operations are based on the different types of volumes created by the user.

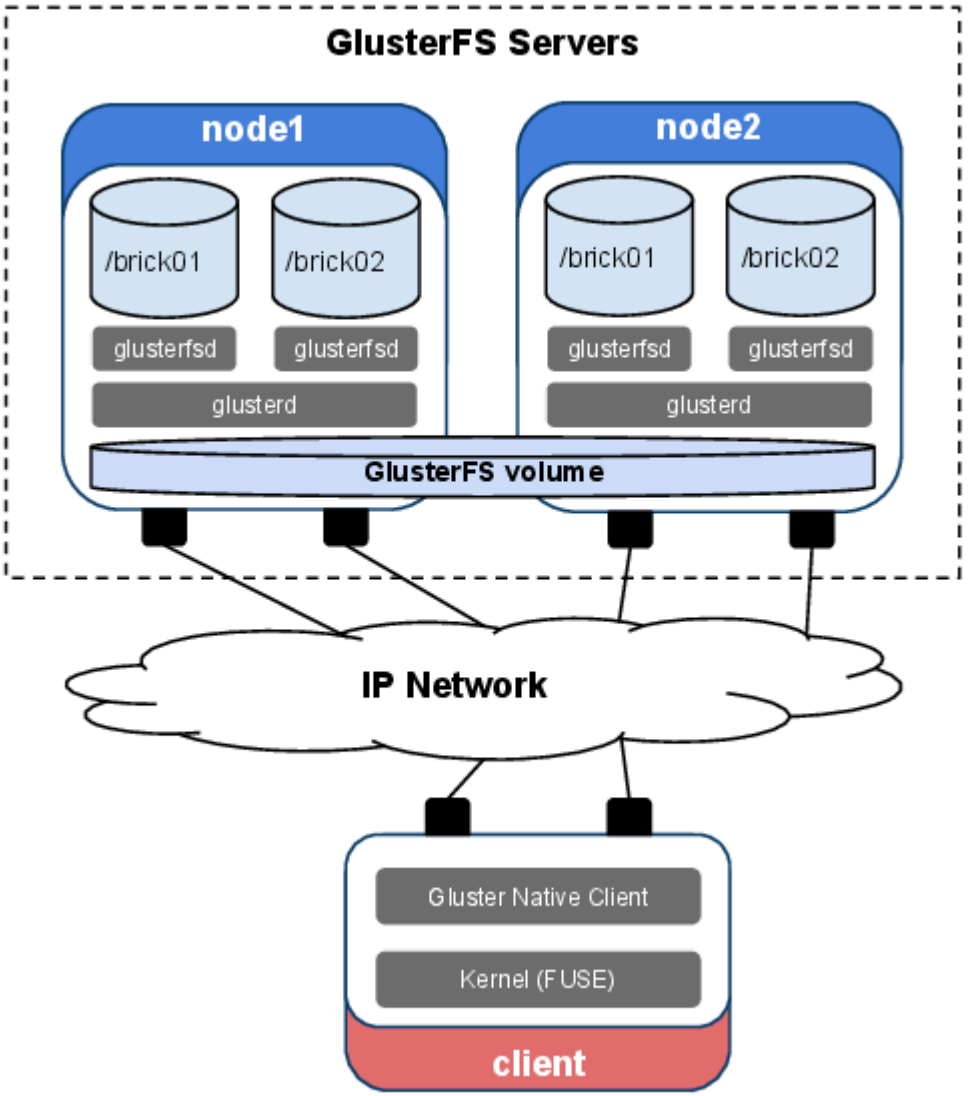
# What is GlusterFS

GlusterFS is a **distributed file system** defined to be used in user space, i.e. File System in User Space (**FUSE**). It is a software based file system which accounts to its own flexibility feature.

Le risorse “memoria e disco” vengono rese disponibili sotto un unico punto di condivisione e tali risorse possono essere montate dai client mediante **tre diversi protocolli**: CIFS, NFS od il client nativo Gluster.

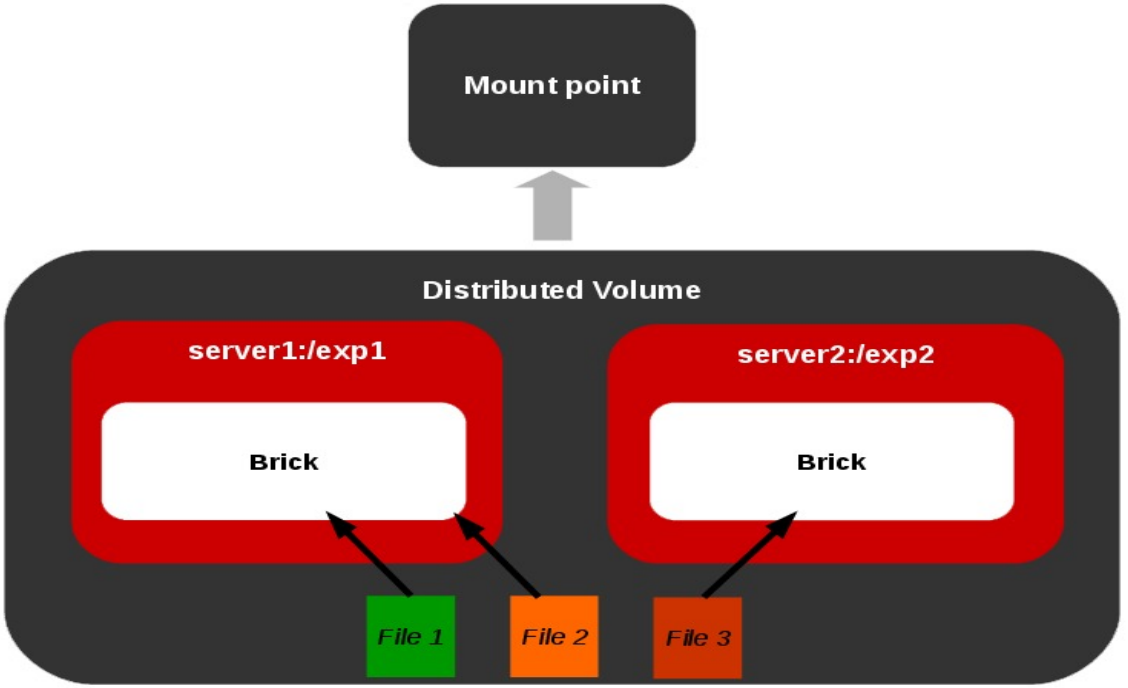
Ligfapi è un ulteriore metodo di accesso a GlusterFS tramite QEMU.

# GlusterFS Architecture



# Types of GlusterFS Volumes: **Distributed**

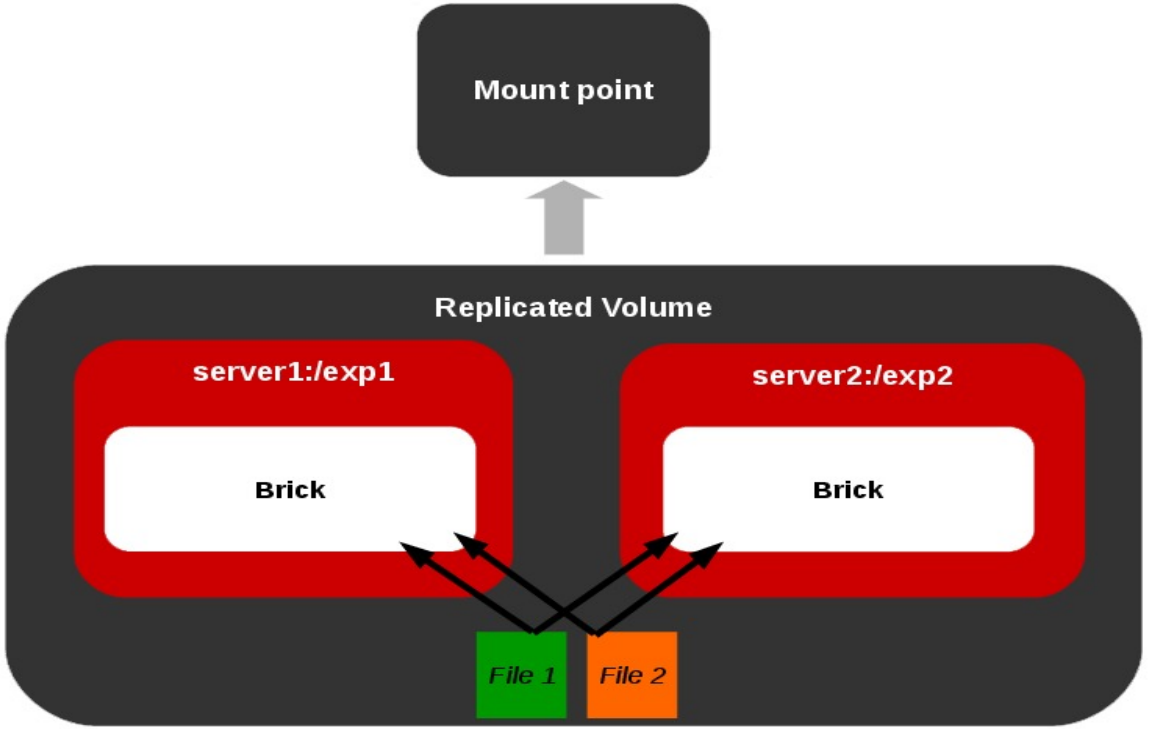
**distribuisce i file all'interno dei brick del volume**




- **Distributed files across various bricks of the volume**
- **Directories are present on all bricks of the volume**

# Types of GlusterFS Volumes: **Replicated**

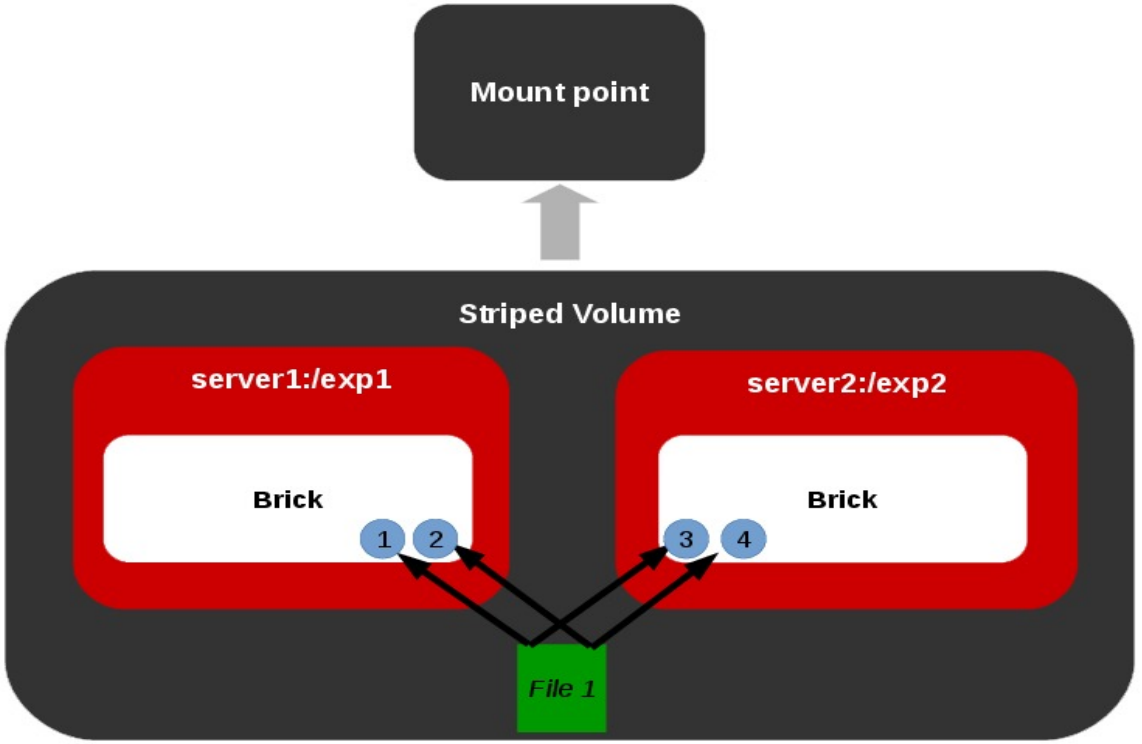
## replica i file nei brick del volume




- **Creates synchronous copies of all directory and file updates**
- **Provides high availability of data when nodes failures occur**
- **Transaction driven for ensuring consistency**

# Types of GlusterFS Volumes: **Striped**

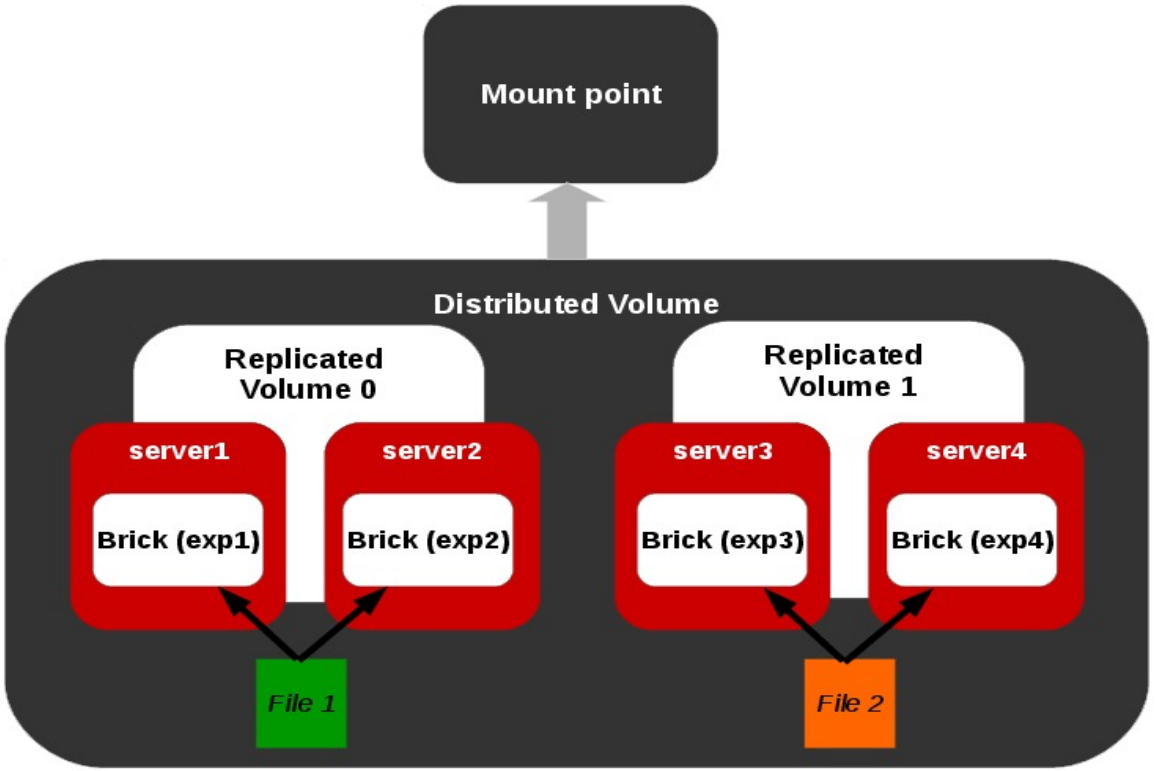
blocchi di dati (stripes) vengono registrati nei brick del volume




- **Files are striped into chunks and placed in various bricks**
- **Recommended only when very large files greater than the size of the bricks**
- **Redundancy with replication is highly recommended since a brick failure can result in data loss**

# Types of GlusterFS Volumes: **Distributed Replicated**

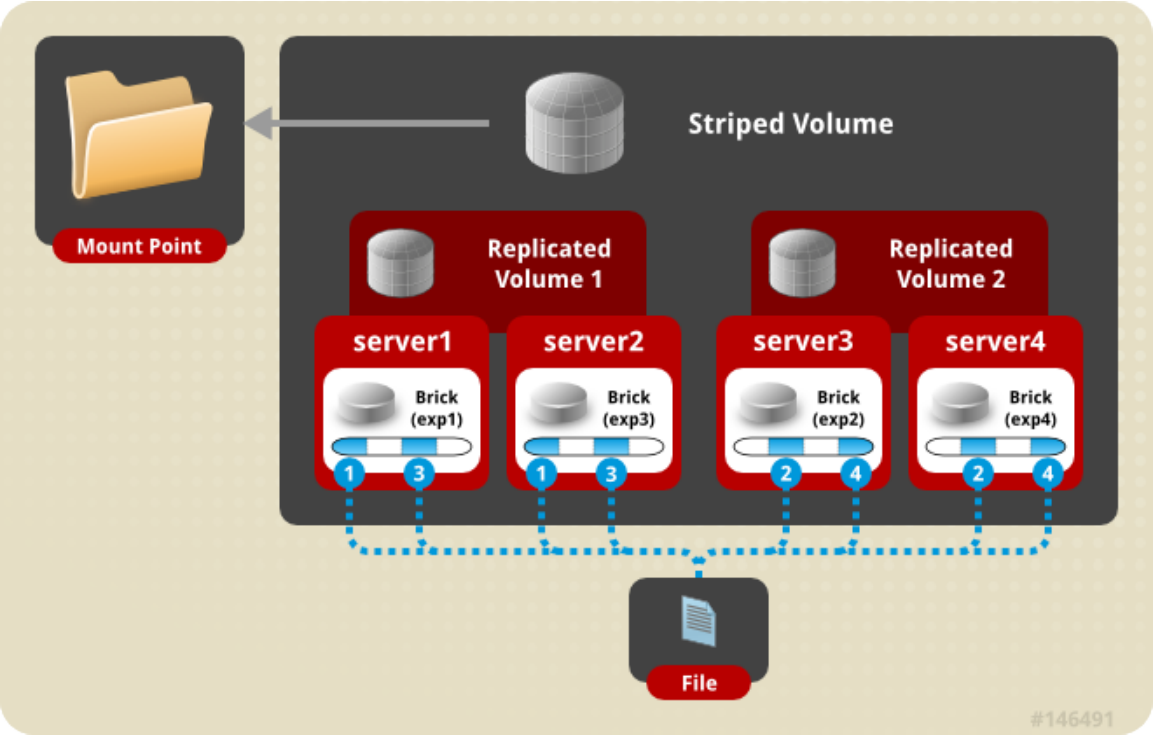
distribuisce i file nelle repliche presenti nei brick del volume



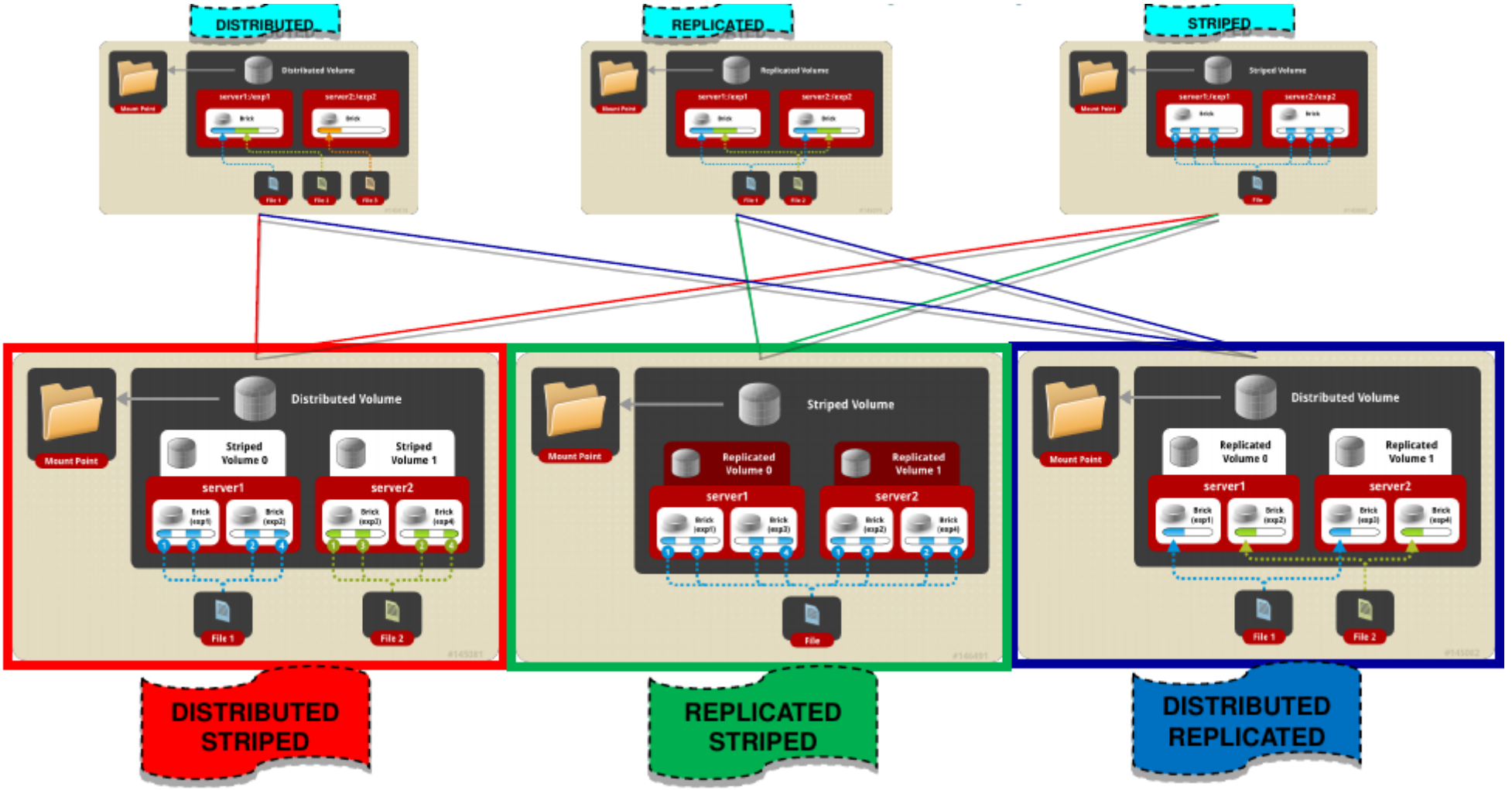
# Different Types of Volumes: **Distributed Striped**

distribuisce i file nei blocchi di dati (stripes) presenti nei brick del volume

# Different Types of Volumes: **Replicated Striped**



# All Type of Volumes



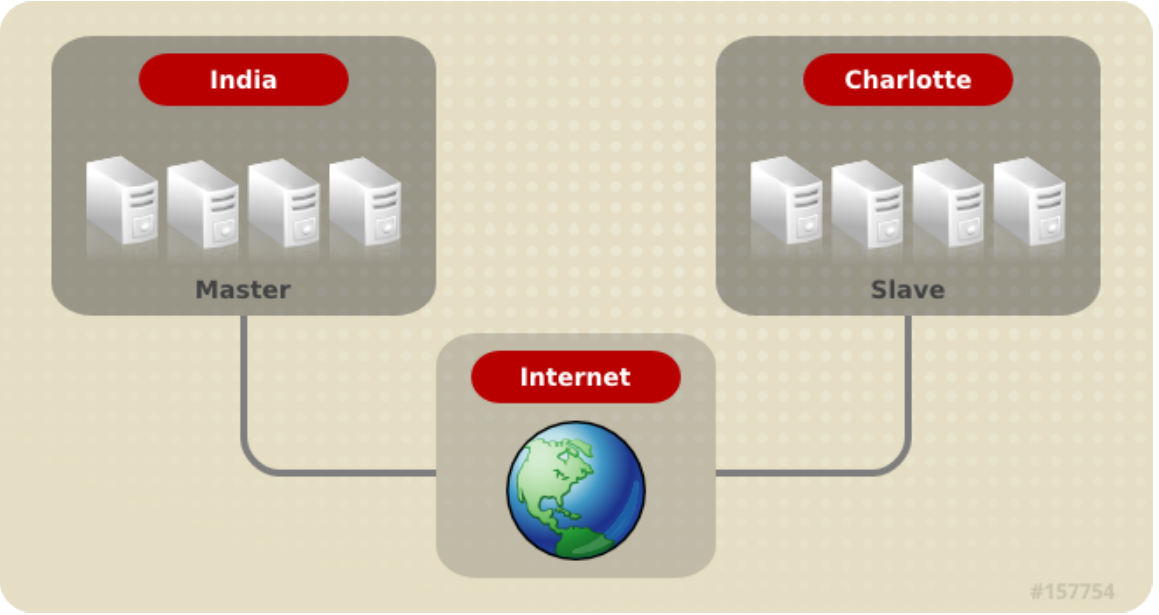
# GlusterFS Volume Example:

**Create a six node distributed and replicated volume with a two-way mirror**

```
$ sudo gluster volume create myvolume \
    replica 2 \
    transport tcp \
    server1:/export/brick/myvolume1 \
    server2:/export/brick/myvolume2 \
    server3:/export/brick/myvolume3 \
    server4:/export/brick/myvolume4 \
    server5:/export/brick/myvolume5 \
    server6:/export/brick/myvolume6
$ sudo gluster volume start myvolume
$ sudo mount -t glusterfs server1:myvolume /mnt/gluster/myvolume
```

# Geo Replication

- Master-slave setup
- Asynchronous incremental replication
- Disaster recovery



# Libgfapi

**Libgfapi** is a POSIX-like C library shipped along with GlusterFS, which allows to access Gluster's volumes without passing through its FUSE client. This integration brings in some benefits but, the most relevant ones are:

- Performance improvements by removing FUSE's overhead.
- Reduce the number of steps required to get to GlusterFS

There's no special configuration needed to use this, as long as you have QEMU >=1.3 and GlusterFS>=3.4 you should be fine. This is an example of what you can do:

•qemu-img create gluster://GLUSTER_HOST/GLUSTER_VOLUME/images5G

# Ovirt GlusterFS Integration

- New feature in oVirt 3.1[1]
- ApplicationMode configuration
  - 1 → Virtualization only (default)
  - 2 → Gluster only
  - 255 → Virtualization + Gluster
- Enable Gluster at cluster level
- New entities (Volumes, Bricks, Volume Options)
- VDSM verbs for gluster management
  - vdsm-gluster plug-in

[1] http://wiki.ovirt.org/wiki/Features/Gluster_Support

# oVirt, Gluster-ized!

- Cluster Management
  - Create Gluster Cluster
  - Add / Remove Storage Servers
  - Delete Cluster
- Volume Management
  - Create Volume
  - Add / Remove bricks
  - Start / Stop volume
  - Delete Volume