Tema 11

Almacenes distribuidos de datos en alta disponibilidad

Profesor : Unai Arronategui
Email : unai@unizar.es

Curso : 2019-20
Grado en Ingeniería Informática
Universidad de Zaragoza

Introducción

- **Objetivo** : proveer diferentes tipos de almacenes de datos persistentes para **alta disponibilidad** (tolerancia a fallos en **escrituras**) y **prestaciones** mediante servicios distribuidos
  - Con NFS, **escritura** de red a **un** solo **servidor**
    - Prestaciones limitadas a hardware de un servidor
      - Mejoras con soluciones **NAS** de altas prestaciones, pero limitación de agregación de ancho de banda de red
        - » Alto coste y complejidad
    - Alta disponibilidad : tolerancia a fallos muy limitada (principalmente lecturas)
  - Otros sistemas de ficheros distribuidos limitados por metadatos :
    - Gestión de metadatos centralizada
- Al menos, un par de soluciones disponibles : **Ceph y GlusterFS**

Tipos de almacenes básicos

- Almacenes de objetos (ejemplo, S3 de Amazon)
  - Datos agrupados en un conjunto de “objetos” al mismo nivel
- Almacenes de bloques
  - Organización de datos en bloques de tamaño
  - Operaciones entrada/salida con dispositivos de bloques
    - Discos duros (o SSDs), particiones, volumenes lógicos, volumenes de red,...
- Sistemas de ficheros
  - Organización jerárquica de ficheros

Consideraciones para administración

- Aspectos relevantes
  - Separación de datos y metadatos
  - Fallos
  - Ocupación de espacio
  - Prestaciones
  - Arquitectura de nodos
  - Arquitectura de red

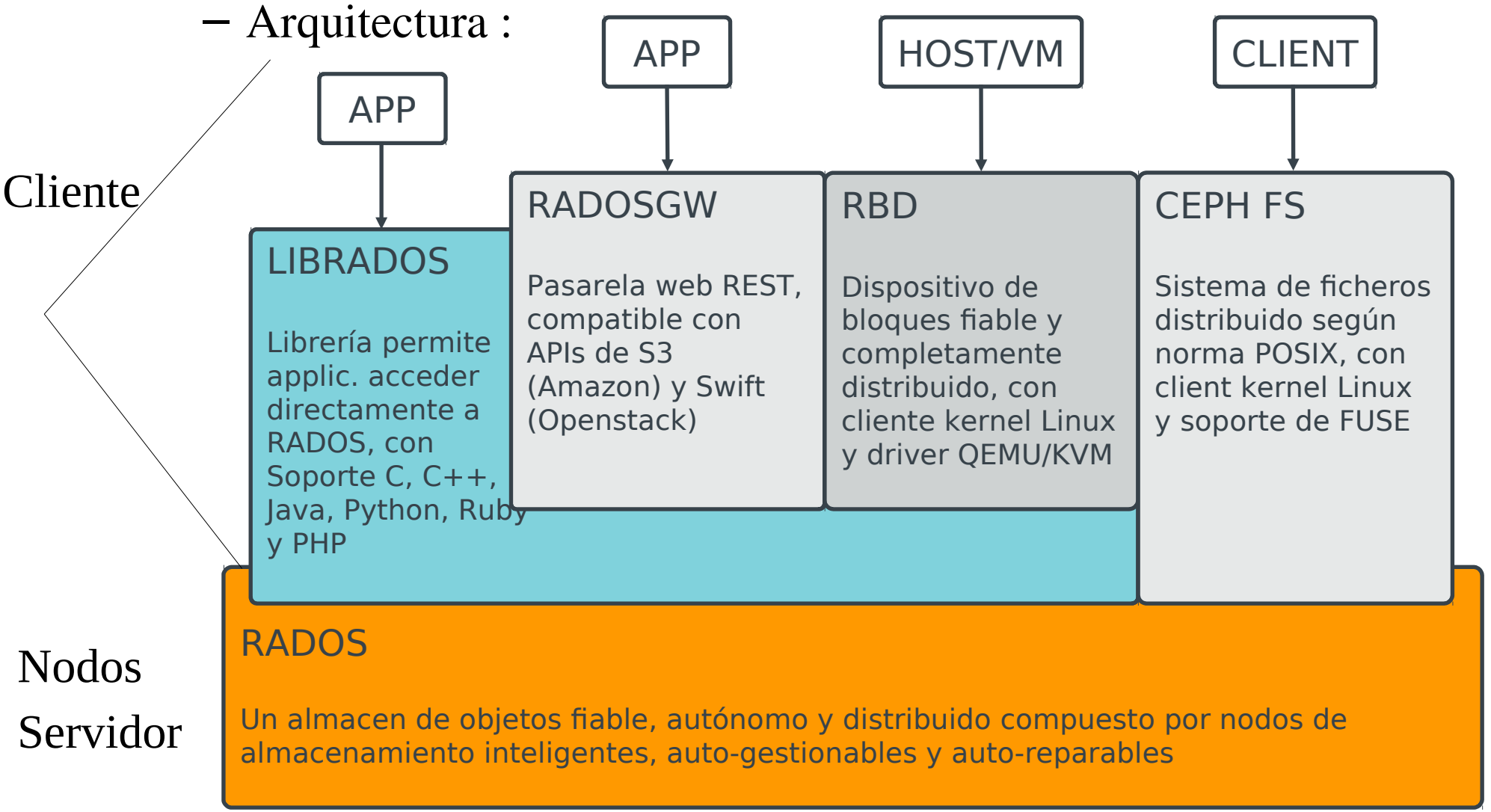
- Tareas
  - Definición del modelo de uso
    - Prestaciones, disponibilidad, prestaciones+disponibilidad
  - Balanceo de datos de usuario entre nodos
    - Automático y/o manual
  - Monitorización
    - Fallos, capacidad, prestaciones, balanceo
    - Automático y/o manual

- Tareas
- Definición del modelo de almacenamiento
  -
  - Balanceo de datos entre nodos
    - Automático y manual
    - Red exclusiva para balanceo, sino puede ahogar acceso de clientes
  - Monitorización
    - Automática y manual

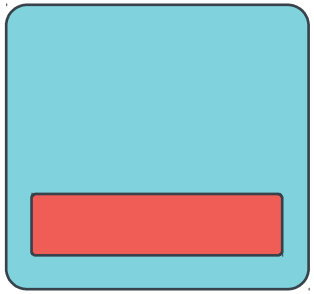
Ceph

- Plataforma unificada de almacenamiento **Ceph** (Linux):
  - Almacenamiento integrado de objetos, bloques de datos y sistema de ficheros distribuidos hasta 16 exabytes. Utilización plataformas cloud.
  - Características :
    - Cada componente debe escalar en número de nodos (< 10.000).
      - Arquitectura distribuida y escalable de servidores de metadatos (Sist. De fich.)
    - No hay ningún punto de fallo.
    - Solución basada en software utilizable en PCs corrientes.
    - Incorporar gestión automática al máximo.
      - Utilización de un algoritmo automático de colocación de datos (**CRUSH**)
  - Casos de uso :
    - Imagenes VMs, Copias de seguridad (datos como objetos de almacenamiento)
    - Como dispositivo de bloque (/dev/rbd/rbd/home)
    - Como Sistema de Ficheros Distribuido (compatible Posix).

– Arquitectura :

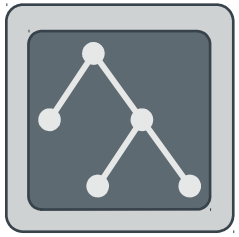


- Dos elementos (demonios) básicos en su funcionamiento :
  - Object Storage Daemon (OSDs) : almacenan datos como objetos en nodos de almacenamiento.
    - Sistema mínimo debe tener, al menos, 2 OSDs para replicación de datos y mínimo recomendado de 3. 10s o 10000s en cluster.
    - Recomendado uno por disco.
    - Cooperan para realizar operaciones de mantenimiento de forma óptima.
      - » Replicación (por defecto 2)
      - » Recuperación
      - » Rebalanceado
      - » Monitorización (detección de fallos en otros nodos de almacenamiento)
      - » Rellenado

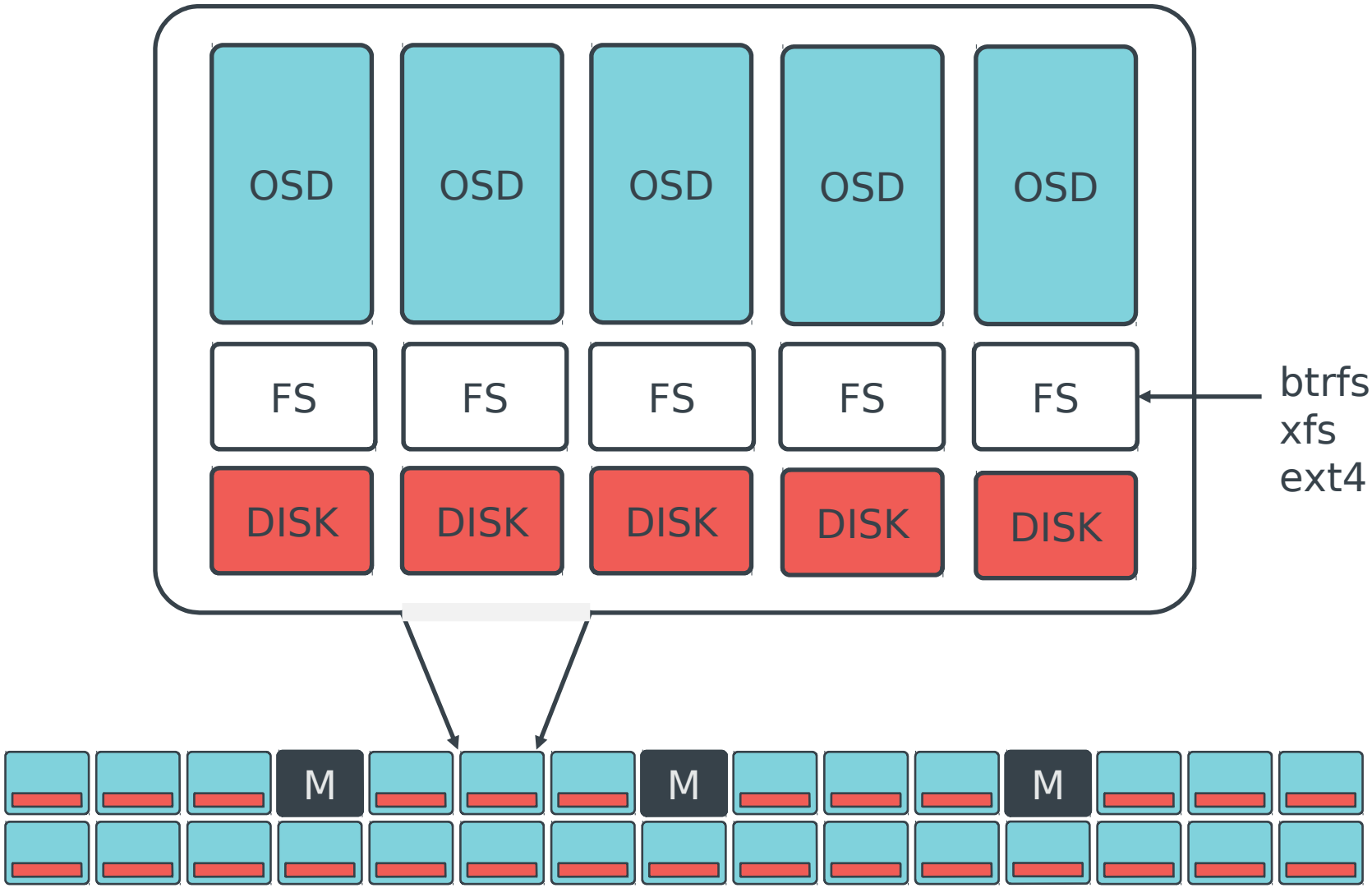


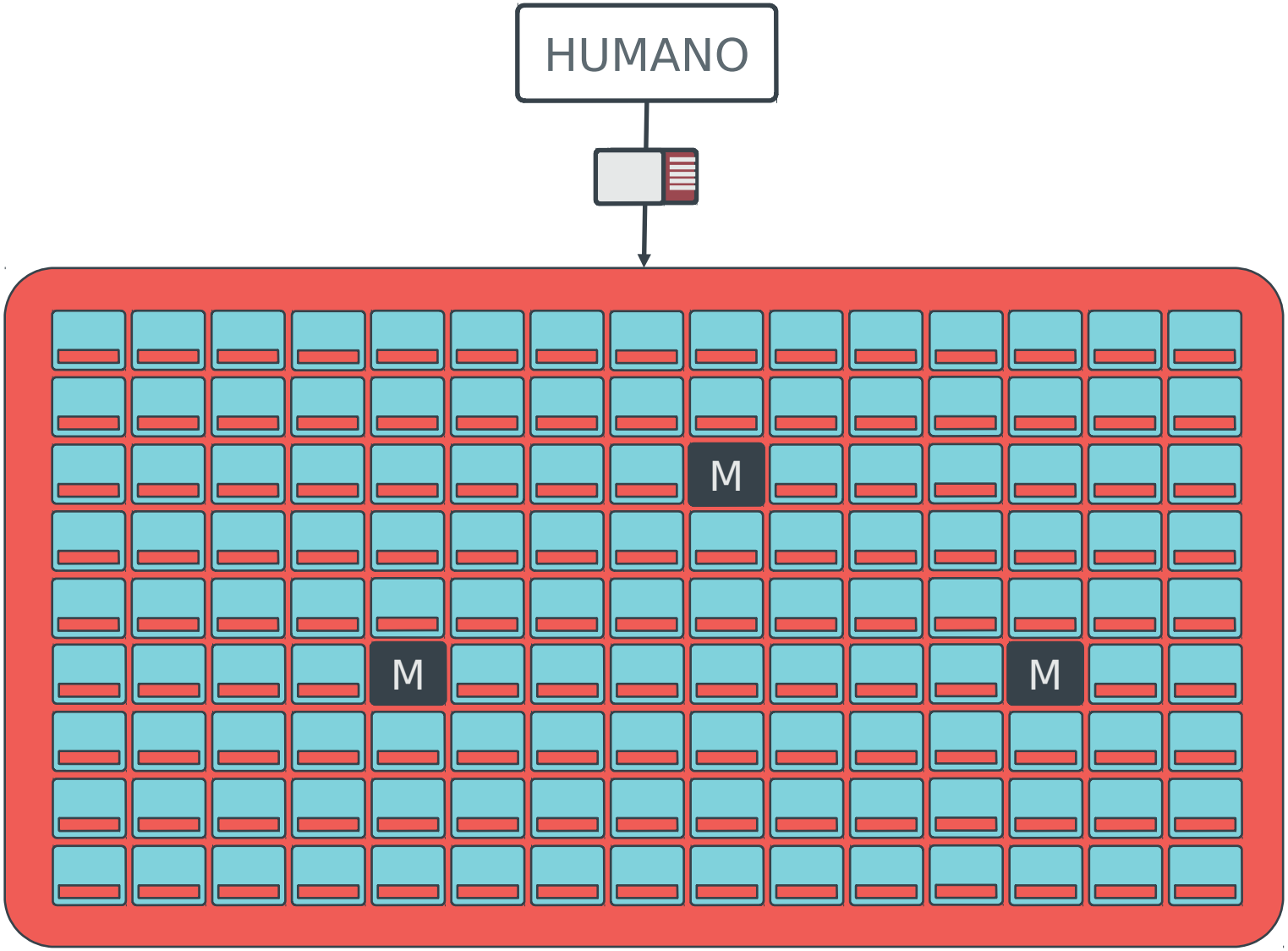
  - Monitores : coordinación **al menos 3**
    - Gestiona pertenencia a cluster y estado
    - Mantienen una copia maestra de mapas del cluster : mapa de monitorización, mapas de OSDs, mapa de grupos de ubicación y mapa CRUSH
    - Proveen consenso para decisiones distribuidas de estos datos (similar a Paxos).
    - No proveen datos a los clientes.

- Un elemento más si se usa como Sistema de Ficheros Distr. :
  - Ceph Metadata Server (MDS) : almacenan metadatos solo para el Sistema de ficheros Ceph :

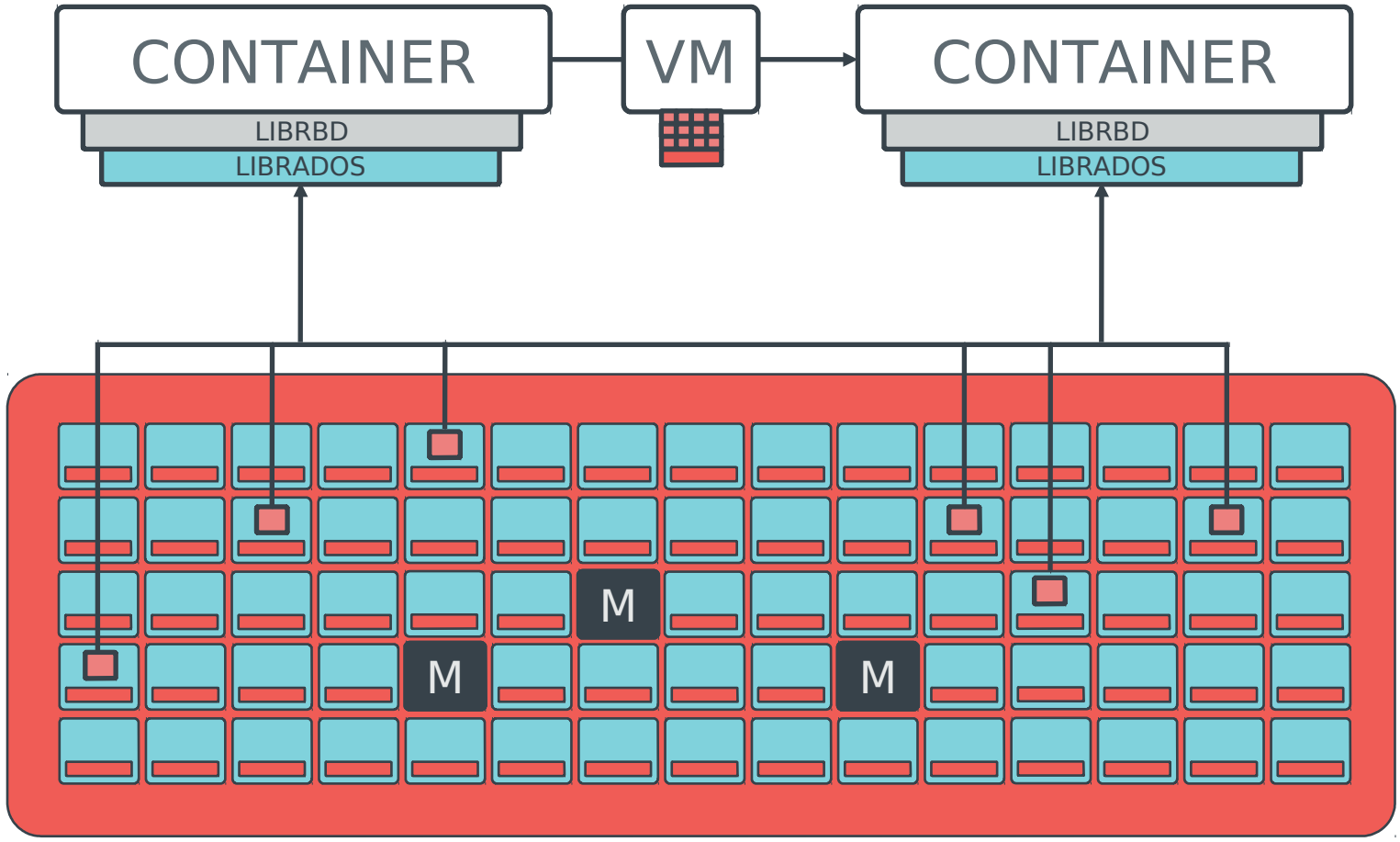


    - No se usa para Almacenamiento de Objetos ni Dispositivo de Bloques
    - Permite realizar las operaciones de fichero de tipo POSIX (ls, find,..), sin sobrecargar el cluster de almacenamiento de Ceph.

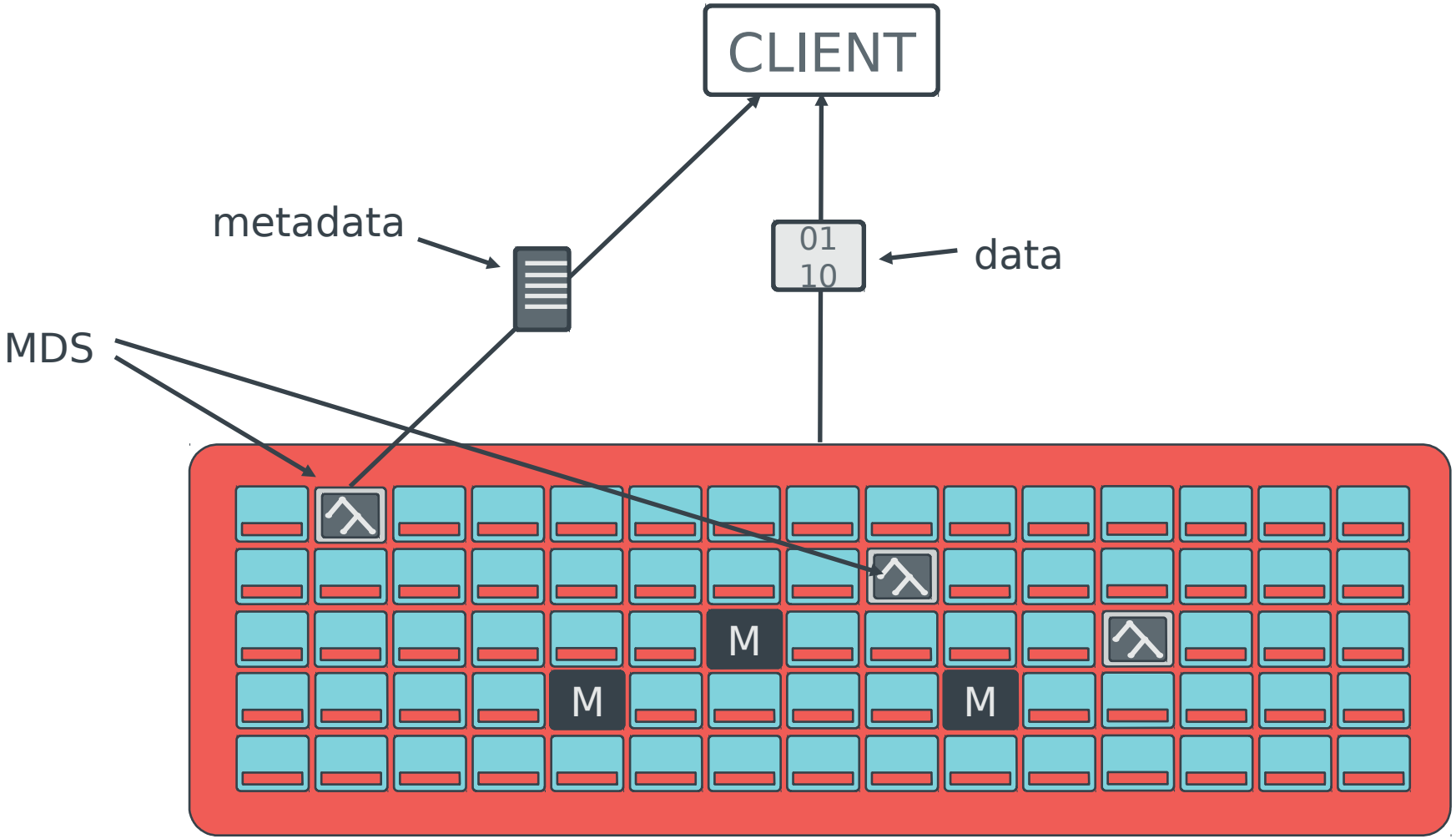

OSD
OSD
OSD
OSD
OSD
FS
FS
FS
FS
FS
DISK
DISK
DISK
DISK
DISK
btrfs
xfs
ext4
M
M
M


HUMANO
M
M
M

- Ejemplo uso 2 dispositivos bloques en una VM :



- Uso como sistema de ficheros :

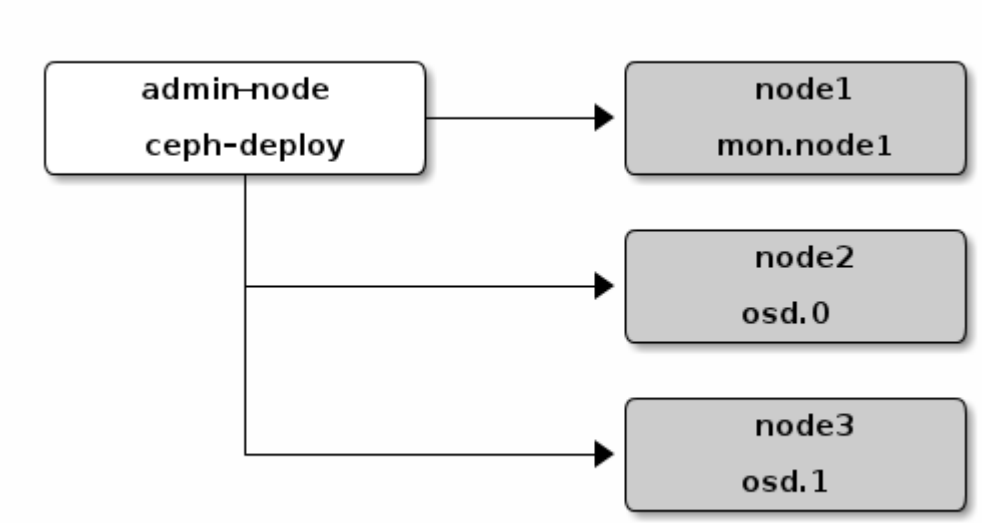


Puesta en marcha de Ceph

- Planificar previamente diseño de red y despliegue de tipos de nodos :
  - Ideal : 3 subredes para el cluster(3 tarjetas por nodo)
    - Acceso clientes (subred pública)
    - Funcionamiento interno del cluster (subred ceph)
    - Acceso administrativo (subred administrativa) : monitorización externa,....
  - Tipos de nodos
    - Tener en consideración que nodos *monitor* usan Paxos :
      - Definir grado de fallos (nº de nodos) para quorum → nº total monitores
    - Definir grado de replicación de datos (*OSDs*) → nº de réplicas por objeto y/o fichero
    - Definir grado de striping de objetos y/o de ficheros → nº de nodos en striping

- Hardware
  - CPU : nodos MDS (> quad core) > nodos OSDs (> dual core) > nodos monitor
  - RAM :
    - Nodos monitor : > 1GB, para disminuir latencia de respuestas
    - Nodos OSDs : > 500MB por 1 TB e almacenamiento por demonio OSD
      - » Más cuando recupera o rebalancea datos entre réplicas
    - Nodo MDS : > 1GB por demonio
  - Red : Al menos 2 tarjetas de red, subred cluster 10Gb ethernet si posible (replicación 1 TB datos con 1Gb/s = 3 horas).
    - Racks (armarios) con conectividad 40-100 Gbps
    - Si se puede, otra subred más para administración out-of-band
  - Dispositivos de almacenamiento
    - Un disco (>= 1TB) dedicado por demonio OSD
    - Discos separados para journal y datos
    - Utilización de SSDs para journals y para metadata pool

- Configuración inicial de ejemplo : 1 nodo administración y 3 nodos para cluster almacenamiento (1 monitor y 2 osd)
  - Cuando este estabilizado (active +clean), se puede añadir 2 nodos monitores adicionales, un servidor MDS (Sfs) y demonios OSD adicionales



  - Para la prueba, el nodo administración pued ser usado como nodo de almacenamiento
  - Cualquier distribucion de Linux. SFs recomendado : *btrfs*.

- Preparar nodos
  - Instalar paquete **ceph-deploy** en nodo administrador
  - Habilitar acceso ssh sin contraseña desde nodo administrador a cada uno de los nodos Ceph
  - Crear elementos iniciales de cluster en nodo administrador
    - **Ceph-deploy –cluster c_prueba new node1** # no hay que ejecutar con privilegios root
      - Define a *node1* como nodo monitor inicial del cluster
      - Crea los ficheros *ceph.conf, ceph,log y ceph.mon.keyring* en directorio en curso del nodo admnistrador.

- En nodos cluster ceph (node1, node2, node3) :
  - Crear usuario asociado en cada nodo ceph

    ```
    ssh usuario@servidor-ceph
    sudo useradd -d /home/ceph -m ceph
    sudo passwd unocomplicado
    ```
  - Añadirle privilegios sudo a nuevo usuario

    ```
    echo "ceph ALL = (root) NOPASSWD:ALL" | sudo tee /etc/sudoers.d/ceph
    sudo chmod 0440 /etc/sudoers.d/ceph
    ```
  - Configurar acesso ssh sin contraseña desde nodo administrador a todos los nodos del cluster ceph (no usar ni sudo ni root)
    - Habilitar utilizar el usuario ceph an accesos ssha cada nodo. En ~/.ssh/config

      ```
      Host nodeX
         Hostname nodeX
         User  ceph
      ```
  - Asegurar conectividad con ping utilizand nombres cortos DNS (sin subdominio). DNS (/etc/hosts ?) y cortafuegos deben estar bien configurados.

- Crear el cluster
  - Definir nº réplicas por defecto a 2
    - En sección [default] de ceph.conf : *osd pool default size = 2*
  - Se aconseja definir, mínimo 2 subredes, una para acceso cliente y otro para tráfico entre nodos cluster (heartbeat, replicación, eetc)
    - Definir la subred pública en sección [global] de ceph.conf

      public network = {@ip}/{mascara_red}
  - Instalar ceph en nodos del cluster :

    ceph-deploy install admin-node node1 node2 node3
  - Añadir monitor(es) inicial(es) y recuperar claves
    - En este caso, inicializa la configuración del monitor en node1

      ceph-deploy mon create

- Añadir 2 OSDs (node2 y node3). Por sencillez, exportaremos directorios en lugar de discos
  - Crear directorio para demonio OSD en cada nodo

    ssh node2 sudo mkdir /var/local/osd0

    ssh node3 sudo mkdir /var/local/osd1
  - Preparar OSDs desde nodo administración

    ceph-deploy osd prepare node2:/var/local/osd0 node3:/var/local/osd1
  - Finalmente, activar los OSDs

    ceph-deploy osd activate node2:/var/local/osd0 node3:/var/local/osd1

- Copiar fichero configuración y clave de administrador a todos los nodos para simplificar administracion posterior con comando “ceph”

  ceph-deploy admin admin-node node1 node2 node3

- Asegurarse que se dispone de permisos de lectura a fichero de clave

  sudo chmod +r /etc/ceph/ceph.client.admin.keyring

- Probar el estado del cluster

  ceph health

  - Si todo va bien, debería devolver : active + clean

- Expandir cluster con un OSD y un MDS en node1 y un monitor en node2 y node3 para establecer quorum.
  - Añadir OSD

    ssh node1 sudo mkdir /var/local/osd2
    ceph-deploy osd prepare node1:/var/local/osd2
    ceph-deploy osd activate node1:/var/local/osd2

    - Una vez añadido, Ceph empieza a rebalancear el cluster (observar con "*ceph w*")
      - Se ve pasar el estado "active +clean" a solo "active" con información de objetos degradados hasta que vuelve a "active + clean" una vez que termina.
  - Crear servidor de metadatos (MDS) para Sistema de ficheros Ceph

    ceph-deploy mds create node1
  - Añadir monitores : 2 más para tener alta disponibilidad con Paxos

    ceph-deploy mon create node2 node3

    - Una vez añadidos, Ceph empieza a sincronizarlos y formar quorum. Test :

      ceph quorum_status --format json-pretty

- Ejemplo de almacenamiento y recuperación de un objeto
  - Cliente debe poner nombre de objeto y "pool" donde está.
  - Con el comando "rados put", utilizamos un fichero como objeto y los ubicamos en el pool "datos"

    rados put test-objeto-1 fich_test.txt --pool=datos
  - Verificamos que lo ha almacenado

    rados -p datos ls
  - Identificar la localización del objeto

    ceph osd map datos test-objeto-1
    - Salida posible :

      osdmap e537 pool 'datos' (0) object 'test-objeto-1' -> pg 0.d1743484 (0.4) -> up [1,0] acting [1,0]
  - Suprimir objeto

    rados rm test-objeto-1 --pool=datos

- Manipulación del cluster Ceph
  - Monitorización del cluster
    - http://docs.ceph.com/docs/master/rados/operations/monitoring/
    - http://docs.ceph.com/docs/master/rados/operations/monitoring-osd-pg/
  - Manipulación de cada tipo de acceso
    - Dispositivo de bloques
      - http://docs.ceph.com/docs/master/start/quick-rbd/
    - Sistema de ficheros
      - http://docs.ceph.com/docs/master/start/quick-cephfs/
    - Almacenamiento de objetos
      - http://docs.ceph.com/docs/master/start/quick-rgw/

- Referencias
  - GlusterFS :
    - http://gluster.readthedocs.io/en/latest/Administrator%20Guide/
    - https://access.redhat.com/documentation/en-us/red_hat_gluster_storage/3.2/html/administration_guide/
  - Ceph :
    - http://docs.ceph.com/docs/master/
    - https://access.redhat.com/documentation/en-us/red_hat_ceph_storage/2/html/administration_guide/

GlusterFS

- Sistema de ficheros distribuido que provee :
  - Escalado lineal (pero limitado a un máximo de 70 nodos, por ahora)
  - Sobrecarga pequeña
  - Alta redundancia
  - Despliegue más sencillo y barato.
  - Composición de prestaciones y alta disponibilidad.
  - Gestión dinámica de distribución de datos
    - Ejemplo : distribución automática de datos a un nuevo nodo

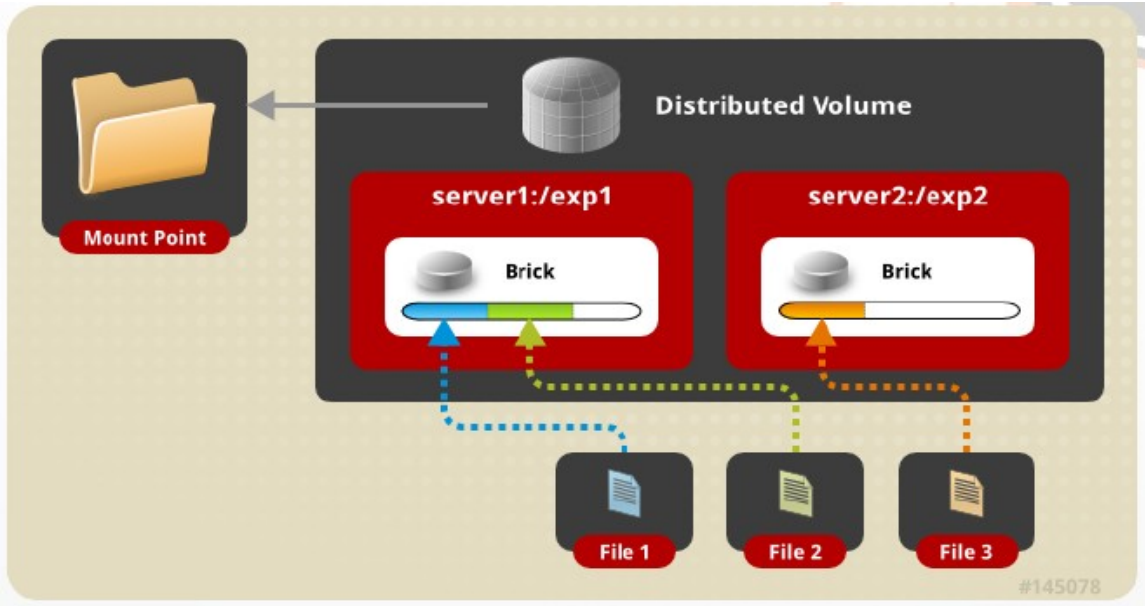
- Casos de uso :
  - Imagenes de máquinas virtuales
  - Computación de altas prestaciones
    - Ancho de banda de acceso a almacen de datos.
  - Componente de almacenamiento para Infraestructura Como Servicio en Cloud.

- Terminología :
  - *Brick* : Unidad de almacenamiento básico de GlusterFS (disco, partición, volumen lógico)
  - *Volumen* : Combinación de Bricks que son exportados meidante traductores
  - Nodo / *Peer* : Servidor ejecutando demonio Gluster y exportando volumenes
  - *Traductor* :
    - Código que implementa la correspondencia entre los bytes alamcenados y el espacio global de nombres del sistema de ficheros
    - Estratificado para proveer la funcionalidad de GlusterFS

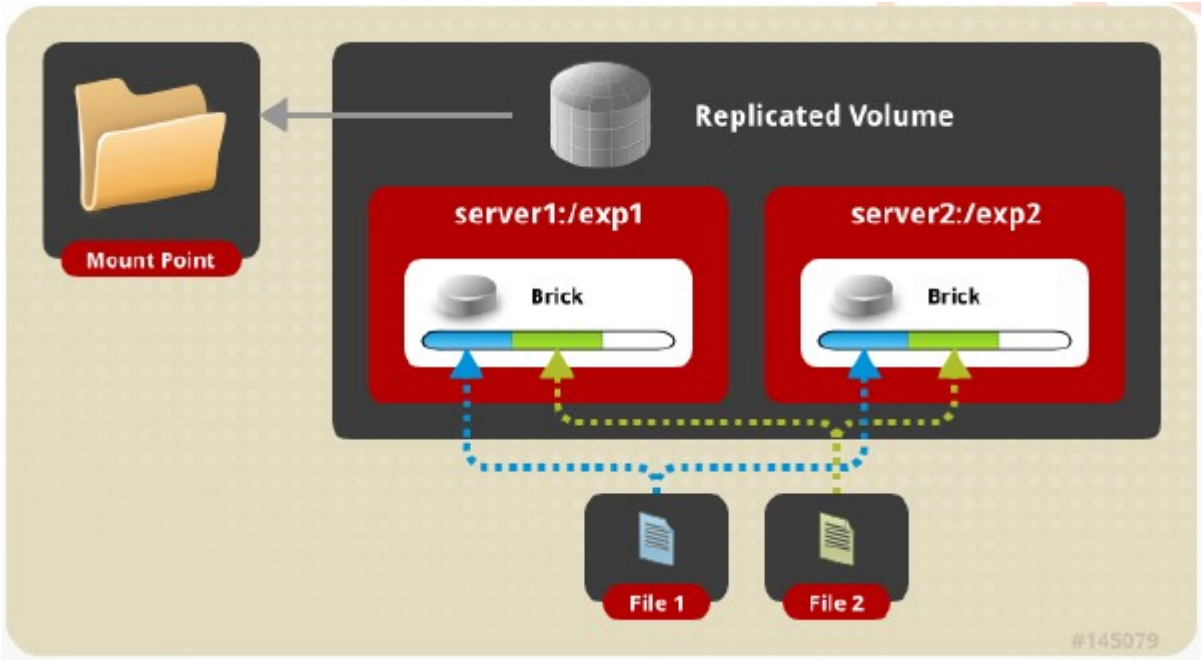
- Componentes :
  - **glusterd**
    - Demonio de gestión elástica de volumenes
    - Prestaciones multi-hilo
    - Se ejecuta en todos los servidores de exportación
    - Gestión mediante línea de comandos
  - **glusterfsd**
    - Demonio de gestión de bricks de GlusterFS
    - Un proceso por brick
    - Gestionado por glusterd

- Glusterfs
  - Demonio servidor NFS
  - Demonio cliente FUSE
- Mount.glusterfs
  - Herramienta de montaje tipo FUSE (ámbito de usuario)
- Gluster
  - Gestión de consola de GlusterFS (línea de comandos)

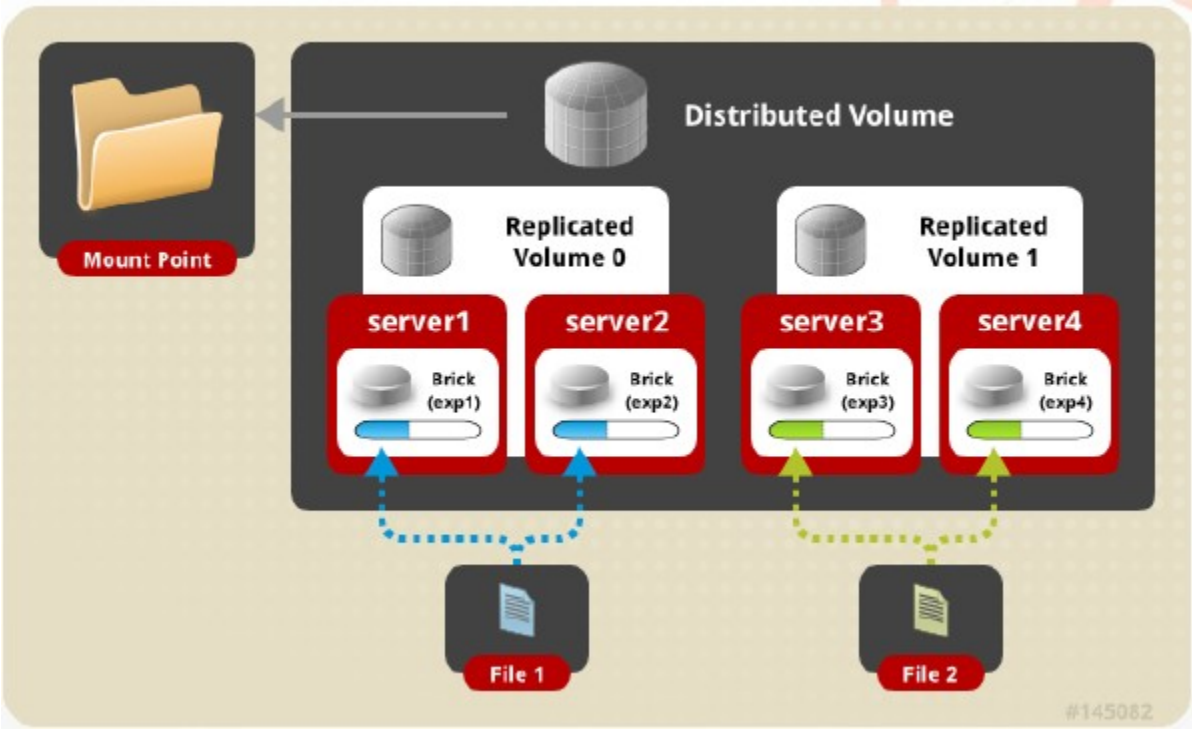
- Volumenes básicos :
  - Volumen distribuido
    - Ficheros esparcidos de forma equitativa entre los bricks
      - Mayores prestaciones en acceso a varios ficheros a la vez
    - NO provee redundancia
    - Ejemplo de uso : imagenes de máquinas virtuales



- Volumen replicado
  - Copia de ficheros de forma redudante mediante replicación síncrona
    - **Consistencia secuencial**.
  - No hay incremento de prestaciones de transferencia de datos

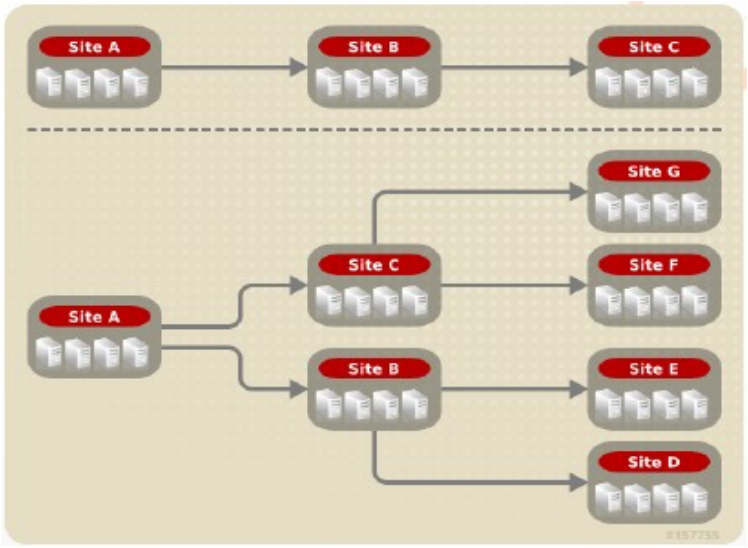


- Volumen Distribuido-Replicado
  - Distribuye ficheros entre bricks replicados
  - Redundancia y prestaciones en acceso a múltiples ficheros



- Volumen trenzado (Striped)
  - Fichero individual repartido entre varios bricks : prestaciones
- Volumen distribuido y trenzado
  - El acceso más rápido a ficheros
- Volumen distribuido, trenzado y replicado
  - La total : rapidez aceso a varios ficheros, a un fichero y tolerancia a fallos
  - Pero necesita un buen número de nodos para obtener esas ventajas
    - Mínimo 8 nodos

- Geo-replicación
  - Gestión **asíncrona** entre LAN, WAN o Internet
  - Modelo Maestro-Esclavo. Posibilidad varios niveles
  - **Consistencia eventual** (continúa, incremental → copias seguridad)
  - Datos son pasados, solo, entre maestro y esclavo emparejados
  - Incremento importante prestaciones con transferencias en paralelo



- Acceso a datos :
  - Cliente Nativo GlusterFS
    - Obtiene datos iniciales de volumen del servidor de montaje y después se comunica directamente con todos los nodos.
    - Recomendado para alta concurrencia y prestaciones en escritura
    - Carga de accesos implicitamente balanceado entre volumenes distribuidos
  - NFS (versión 3)
    - Automontador soportado, gestión de bloqueos incluido, cacheo en cliente
    - Balanceo de carga debe ser gestionado externamente.

- SMB/CIFS
  - Gestión manual en cada nodo, para acceso, balanceo de carga y recuperación IP.
- Ficheros y objetos unificados
- libgfapi
  - Librería acceso a GlusterFS que puede ser integrada en una aplicación :
    - Eliminación de copias, cambios de contexto de Sistema Operativo...
    - Pero la misma semántica de volumenes y traductores de GlusterFS

- Preparación de nodos servidor (Peers)
  - Crear Sistema de ficheros en bricks. En todos los nodos :

```
sudo mkfs.xfs -i size=512 /dev/sdb1
sudo mkdir -p /data/brick1
echo "/dev/sdb1  /data/brick1  xfs  defaults  1  2" >> /etc/fstab
sudo mount -a
```

  - Instalar y arrancar GlusterFS en cada nodo :

```
sudo apt-get install glusterfs-server
sudo service glusterfs start
```

  - Configurar el pool de confianza (conectar los nodos)
    - Desde el primer servidor al resto : **$ sudo gluster peer probe servidor2**
    - Con nombres DNS, es necesario que este primer servidor sea contactado por uno diferente. Desde servidor2 : **$ sudo gluster peer probe servidor1**
  -

- Crear volumen distribuido

```
sudo mkdir  /data/brick1/gv0
sudo gluster volume create gv0 server1:/data/brick1/gv0 server2:/data/brick1/gv0
sudo gluster volume start gv0
sudo gluster volume info
```

- Crear volumen replicado (2 nodos)

```
sudo mkdir  /data/brick1/repgv1
sudo gluster volume create repgv1 replica 2 server1:/data/brick1/repgv1 server2:/data/brick1/repgv1
sudo gluster volume start gv0
sudo gluster volume info repgv1
```

- Puesta en marcha en el cliente

```
sudo apt-get install glusterfs-client
mount  -t  glusterfs  server1:/gv0  /mnt
```

  - Probar si funciona :

```
for i in `seq -w 1 100`; do cp -rp /var/log/messages /mnt/copy-test-$i; done #en el cliente
ls -lA /mnt | wc -l           # en el cliente
ls -lA /data/brick1/gv0    # en los nodos servidor
```

- Herramientas complementarias
  - Heketi (https://github.com/heketi/heketi)
    - Gestión del ciclo de vida de volumenes para GlusterFS
      - Gestión de nodos en zonas de red diferenciadas para gestión de fallos
      - Gestión de múltiples clusters GlusterFS simultaneamente
      - Posibilidad de integración con plataformas cloud (Kubernetes, OpenStack) para provisión automática de nuevos volumenes
      - Protocolo de red RESTful

- Definición de topología de cluster, nodos y discos mediante fichero formato json :

```
{
   "clusters": [
      {
         "nodes": [
            {
               "node": {
                  "hostnames": {
                     "manage": [
                        "10.10.43.61"
                     ],
                     "storage": [
                        "10.10.43.61"
                     ]
                  },
                  "zone": 1
               },
               "devices": [
                  "/dev/sdb"
               ]
            },
            {
               "node": {
                  "hostnames": {
                     "manage": [
                        "10.10.43.153"
                     ],
                     "storage": [
                        "10.10.43.153"
                     ]
                  },
                  "zone": 2
               },
               "devices": [
                  "/dev/sdb"
                ]
            }
         ]
      }
   ]
}
```