SDS
Open Source

Treball Final de Carrera 2014

Autor: Saxa Egea Oliver

Consultor: Ignasi Rius Ferrer

13 de juny de 2014

UOC
Universitat Oberta de Catalunya

Agraïments

Arribats aquest punt de la carrera em veig obligat a agrair l'ajuda rebuda al llarg de tots aquests anys:

El Josep que tant em va ajudar amb les assignatures de matemàtiques i estadística.

El Vicente que gràcies a les seves lliçons magistrals vaig entendre la programació orientada a objectes.

La família que m'han donat tot el suport i els ànims del món en tot moment.

I en especial a la meva dona l'Anna, qui m'ha permès aquests dos últims meravellosos anys, dedicar temps i esforços a enllestir aquesta cursa contra-rellotge.

I també vull demanar disculpes a la meva filla, la Naia, per no haver pogut estar tot el temps que voldria amb ella.

Moltes gràcies a tots i totes per ser-hi i per com sou!

Índex

Introducció

El Treball de Fi de Carrera és la culminació de molts anys de dedicació i esforços per a superar l'Enginyeria Tècnica en Informàtica de Sistemes i ha de servir per mostrar l'assoliment dels coneixements adquirits.

Degut a la meva feina i per preferències personals he volgut posar en marxa un projecte que està en plena explosió: el Software Defined Storage (SDS). Al llarg d'aquest treball mostraré diferents opcions d'implementacions basades en programari lliure. Les compararé entre elles i, finalment, amb les diferents opcions comercials dels principals fabricants de maquinari dedicat a l'emmagatzemament.

És un projecte ambiciós degut a la complexitat que implica. Per a posar en marxa el pilot cal preparar un desplegament de sistemes força complexe, un nombre relativament alt d'equips (que en el meu cas ho faré mitjançant virtualització) i els serveis necessaris per a fer-los funcionar.

Però el pilot només serà la culminació d'una avaluació de l'estat actual d'algunes de les diferents tecnologies disponibles. Podrem veure com posar en marxa un projecte que, a petita escala, mostrarà els avantatges i desavantatges d'un Software Defined Storage.

Finalment miraré d'obtenir conclusions amb l'ajuda d'un repàs general del que està fent les grans marques comercials com ara fabricants com Hitachi, EMC i NetAPP.

Motivació

Periòdicament tota empresa que creix ha de saltar d'esglaó tecnològic. I el cas de l'empresa on treballo no és una excepció. Des de fa un temps estem enmig d'aquest nou esglaó o canvi tecnològic pel que fa a l'emmagatzemament. Aquest és un cicle més que es presenta força complicat. Els requeriments de l'empresa han evolucionat fins a unes dimensions que fa pocs anys eren inimaginables: Actualment ja tenim requisits de més de tres-centes Terabytes i l'objectiu a quatre anys que tenim en ment és la magnitud d'un Petabytes i l'empresa té uns nivells d'exigència que ja justifiquen els sistemes redundants i disposar de plans de Disaster Recovery (DR).
Això planteja una sèrie de dubtes força complexes de respondre: Quan arribi el proper cicle, com ens plantejarem la migració de les dades a una nova *commodity*? Podem mantenir un sistema de còpies de seguretat amb aquest dimensionament? El cost d'implantació, és proporcional a la magnitud?

Aquests dubtes fan replantejar-se les solucions propietàries clàssiques. I és en aquest punt on els Software Defined Storage (SDS[1]) prenen més rellevància. Aquests SDS permeten aïllar-se dels fabricants fent servir maquinari de propòsit general (Commodity hardware), ergo això ens ha de permetre arribar a una facilitat per a incorporar maquinari de diferents fabricants i no haver de fer complicades migracions de dades.

Els nous paradigmes del "as a Service" ja posen el Datacenter as a Service com un objectiu per a l'evolució de les empreses. Els SDS han de ser la base per a un sòlid creixement i implementació del DCaaS, passant per l'Infraestructure as a Service (IaaS) com vSphere o OpenStack, i el Platform as a Service (PaaS) com Openshift.

1 SDS (http://en.wikipedia.org/wiki/Software-defined_storage)

Objectiu

L'objectiu principal d'aquest treball és el d'avaluar algunes de les opcions més rellevants del món Open Source que hi ha al mercat per al emmagatzemament com a servei o per programari.

Analitzaré els requeriments d'una empresa per tal de poder definir uns criteris d'avaluació de les diferents opcions.

Les solucions verticals, o Scale Up, són aquelles que es basen en fer créixer el maquinari incorporant-hi més discos, processadors, memòria, etc. Aquestes solucions són les de més ràpida implantació i augment de capacitats, però l'augment de capacitat o potència estan limitades a la tecnologia i els models existents. Arribats aquest punt cal canviar el maquinari per un de superior, incorrent en costoses operacions (temps, per tant, diners) i aturades de servei.

Les solucions de creixement horitzontal, o Scale Out, són les que distribueixen les tasques entre diferents nodes o particions d'un ens total. En els SDS les solucions permeten el creixement fins a límits que probablement no assolirem mai, o, en el seu defecte, les actualitzacions de les versions, incrementaran per a no tornar a arribar-hi.

El Big Data i la virtualització sobre núvols (clouds) cada dia requereixen de més funcionalitats i més capacitats. Projectes com Openstack cada dia estan prenent més força i, sota d'ell, requerix un emmagatzemament que serveixi com una bona base.

Mostraré algunes de les capacitats més interessants que aporten aquestes solucions Open Source al món del *storage*.

Estructura del document

El present document es desenvolupa al llarg de tres fases ben diferenciades.

En el primer apartat analitzarem les necessitats i requeriments que una empresa qualsevol pot tenir davant les infraestructures d'emmagatzemament. Està basat sota les necessitats d'una empresa real però pot ser extrapolable a qualsevol empresa d'una grandària mitja o superior.

Posteriorment miraré d'analitzar algunes de les solucions Open Source més incipients sobre el concepte Big Data i Scale Out. I crearé un pilot sobre una d'aquestes solucions intentant aconseguir complir tots els requeriments i necessitats que he analitzat al primer punt.

Finalment miraré d'exposar les conclusions que es poden obtenir a partir d'aquest pilot, i intentaré aproximar-ho a les solucions propietàries que, a data d'avui, són les que disposen de més recursos per al seu desenvolupament, i, tanmateix, val a dir que avui són també les més esteses a nivell empresarial.

Anàlisi de requeriments

Per analitzar els requeriments ens basarem en un exemple real d'una gran empresa, una empresa multinacional amb més de dos mil treballadors amb presència a més de cinquanta països i amb seus productives a vuit d'aquests.
Tot i que només sigui un cas concret la seva estructura no és tant diferent a la d'altres empreses. La casuística potser està portada a l'extrem, però és un bon exemple del que passa a moltes altres empreses. Així, tot i que d'altres empreses no tinguin tots els següents requeriments, segurament ajudarà a qui s'estigui plantejant una solució d'aquest tipus.

 Fet: El rendiment és una necessitat.

La concurrència d'usuaris sobre uns serveis corporatius centralitzats provoca que aquests tinguin una demanda de rendiment (IOPS[2]) molt elevada.
Hi ha sistemes que provoquen estrès sobre els discos, com poden ser bases de dades o servidors de correu.

 Fet: No es poden perdre dades.

El sistema ha de garantir la integritat i la persistència de les dades. Cal tenir-les suficientment redundades per a assegurar-ne la seva continuitat.

 Fet: Gran capacitat.

L'elevada capacitat de la informació a emmagatzemar és una altra característica essencial en les grans empreses. A més, aquesta elevada capacitat dificulta els mecanismes de còpies de seguretat, els plans de continuïtat de negoci (BCP) i els de de recuperació davant de desastres (DR).

 Fet: Dispersió geogràfica.

La dispersió geogràfica ens porta a indrets on les comunicacions no són especialment bones. Fins i tot a la seu principal que està ubicada prop de Tarragona, on no hi arriba, a data d'avui, comunicacions en fibra òptica. La deslocalització ens porta a països com la Índia, la Xina o el Brasil on les latències són molt altes degut a filtres en alguns casos o distància en d'altres.

 Fet: Múltiples tipus d'accés a les dades.

Els serveis bàsics de moltes empreses són:
- Correu electrònic.
- Servidor de fitxers.
- Programari de gestió empresarial (ERP).
- Bases de dades corporatives.

A més, degut a les característiques d'aquesta empresa, hi ha algunes aplicacions més específiques com ara:

2 IOPS (http://en.wikipedia.org/wiki/IOPS)

- Clústers de computació per a càlcul intensiu (HPC[3]).
- Aplicacions WEB per a la gestió dels diferents productes del negoci.

La diversitat de serveis i la gran quantitat de servidors es veu alleugerida per la implantació de sistemes de virtualització. No és l'àmbit d'aquest TFC endinsar-nos en aquest món, però l'empresa actualment disposa de dos serveis de virtualització: VMWare i XenServer. En tots dos casos aquests sistemes de virtualització suporten l'ús de storage de tipus SAN i de tipus NAS, aquest últim accedint-hi mitjançant NFS.

 Fet: High Performance Computing

Un dels processos de la empresa és la simulació mitjançant clústers de càlcul intensiu. El departament de simulació, és un cas especialment sensible a dos factors: La capacitat i el rendiment. Es generen quantitats ingents d'informació on només una simulació d'unes poques hores pot significar un fitxer de més de 50GB. El temps que una estació de treball trigarà a obrir un fitxer de resultats d'aquesta grandària va directament relacionat amb l'ample de banda que el sistema de storage és capaç d'entregar i la concurrència que pot assimilar.

 Fet: IOPS

Els sistemes de gestió de bases de dades necessiten un sistema d'alt rendiment i baixa latència degut a l'elevat nombre de consultes que processen. Aquests sistemes comercials requereixen sistemes d'accés a bloc per garantir la coherència de dades que gestiona el propi sistema.

 Fet: Còpies de seguretat

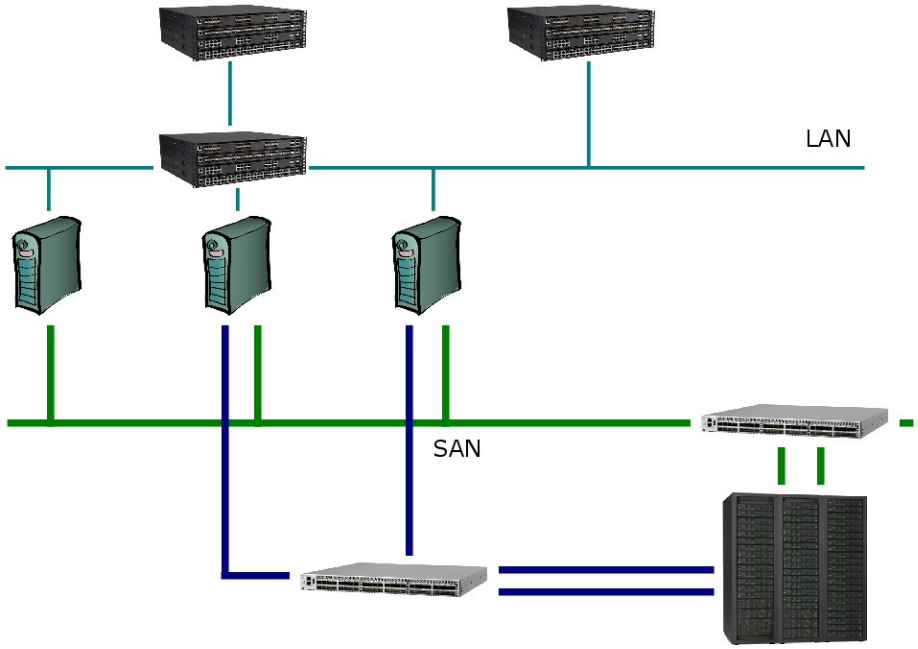
Com ja sabem aquest sistema ha de emmagatzemar una gran quantitat d'informació, fent que les còpies de seguretat tradicionals (sobre cintes o sobre *virtual tapes*) siguin força complexes. Cada cinta que fem servir per a una còpia de seguretat incorpora un punt de fallida que, en cas de desastre, ens pot impedir la recuperació del sistema. D'altra banda disposem de finestres de *backup* que ens marca el període màxim de temps que disposem per a dur-la a terme. No oblidem que els sistemes redundats no eximeixen de disposar de còpies de seguretat ja que no prevenen l'error humà (usuari que esborra un fitxer) o d'aplicacions (una aplicació s'ha parat enmig d'un procés de salvat). En aquest punt, les instantànies o snapshots d'un volum o sistema de fitxers poden actuar com un sistema de backup per aquestes situacions.

Tots aquests serveis ens porten a la necessitat de tenir diferents mètodes d'accés a la informació i caldrà determinar si per diferents canals:

Un esquema de xarxa clàssic fa servir dues xarxes independents: Una basada en tecnologia *Ethernet* i l'altre en tecnologia *Fiber Channel* (FC). La xarxa FC té un cost força elevat: mentre una tarja de xarxa (GE) pot costar menys de 100€, una tarja FC de 4GB pot elevar el seu cost fins a més de 600€. És per això que els servidors més crítics acostuma a posar-s'hi un doble camí cap a diferents switches de FC i d'altres queden amb només un camí. La cabina de discos està connectada a ambdós switches. Els servidors fan servir la capa superior de xarxa Ethernet per a arribar als clients o estacions de treball.

3 HPC (http://en.wikipedia.org/wiki/High-performance_computing)

Figura 1: Esquema LAN/FC

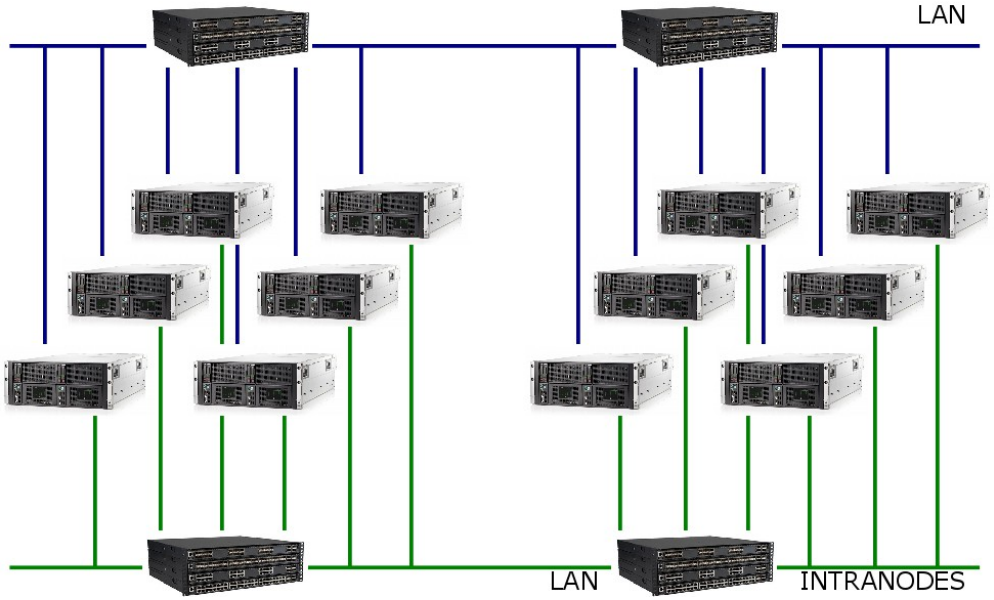


Aquest model ja comença a considerar-se antiquat en moltes instal·lacions actuals. Les connexions FC sembla que estan arribant al final de la seva vida útil donant pas a la xarxa Ethernet fent servir els protocols FC (FCoE) o iSCSI i aprofitant-se de:

- Velocitats de fins a 100Gbps
- Costos reduïts i controlats: Les targes GE i 10GE estan a preus ja força assequibles.
- Possibilitats d'enrutar el tràfic contra diferents centres de dades.

Amb aquest nou escenari la xarxa passa a unificar els seus punts de fallida, tornant-se més crítica però facilitant-ne la seva gestió. Un esquema on hi hagi un clúster podria assemblar-se més a un disseny com el de la figura 2.

Figura 2: Esquema LAN d'un Clúster



La casuística intrínseca que aporten els SDS és que quan manca rendiment per concurrència, el sistema ha de permetre afegir més nodes, amb el que aconseguirem més ample de banda per al sistema i més eixos (discos) per servir la informació, i per tant, més rendiment.

Storage Area Network (SAN)

Storage Area Network va ser la resposta a la problemàtica del Direct Attached Storage (DAS) on cada sistema disposava d'uns discos interns o connectats directament per controladores pròpies. Això repercutia en la gestió i administració diversificada, ja que calia gestionar per separat una gran quantitat de discos independents. Es van unificar aquests discos en una gran cabina de discos que disposa d'una intel·ligència pròpia. Permet definir diferents nivells de RAID[4] que ofereixen nivells redundància i de rendiment específics per a cada sistema, però intenten aprofitar al màxim totes les capacitats. Presenten discos virtuals a cada sistema connectat a la xarxa SAN o LUN.

La SAN requereix d'un *hardware* específic per a connectar cada equip amb el sistema central. Els medis físics a emprar poden ser de diferents tipus:

- Fibra òptica: La més estesa a data d'avui, utilitza aquest tipus de connectivitat. Fa servir un protocol anomenat Fibre Channel Protocol (FCP) que, bàsicament, utilitza les instruccions SCSI sobre un canal de fibra òptica. Les velocitats màximes que estan assolint és de 16 Gbps i, tot i que està projectat els 32 Gbps i 128Gbps, sembla que no acabarà de progressar. Les més esteses són les de 4 Gbps.
- Ethernet: Fent servir el protocol FCoE es canvia el medi físic i es fa passar a través de connexions Ethernet. Les velocitats que s'estan aconseguint actualment són de 10Gbps i de 40Gbps, i s'està treballant en els 100Gbps.

4 RAID (http://en.wikipedia.org/wiki/RAID)

– iSCSI: Encapsular mitjançant trames IP el protocol SCSI. Aquest tipus de connectivitat té el *handicap* de la sobrecàrrega que afegeixen les capçaleres IP a les trames SCSI. Els medis físics emprats també són els de Ethernet. Avui en dia hi ha adaptadors de xarxa especialment dissenyats per alleugerir la sobrecàrrega que implica al processador de l'ordinador el tractament d'aquestes capçaleres IP.

L'accés a bloc no permet l'accés des de servidors al mateix volum o unitat, excepte en alguns casos on el sistema és capaç de gestionar els bloquejos i l'accés concurrent des de diferents servidors. Així l'accés en mode bloc obliga a disposar de rèpliques dels volums a d'altres dispositius o cabines on romandrà la informació sense ser usada, amb una mica de sort, mai. Són els sistemes actius-passius, on un segon sistema està a la espera que hi hagi un desastre per començar a treballar.

Network Attached Storage (NAS)

Mitjançant els protocols d'accés a fitxers (NFS, CIFS, AFP, FTP,...) facilita la homogeneïtzació de les dades, unificant-les en un únic punt. No requereix de cap *hardware* específic. Un dispositiu a la xarxa serveix el seu espai com a punts de muntatge o carpetes compartides, permeten l'accés als mateixos fitxers per diferents protocols. Justament aquest últim punt presenta una problemàtica per controlar qui està accedint al fitxer a cada moment, és el problema de la gestió de bloquejos.

En aquest apartat podrem valorar les diferents opcions que disposa cada producte. Podem valorar si el producte té un client específic instal·lable a les diferents plataformes que hi pugui haver en el nostre parc, o per contra hem d'aconseguir que el nostre clúster "parli" els protocols necessaris per a cada client.

També és de vital importància poder definir llistes d'accés als directoris (ACLs). Un punt clau a tenir en compte d'aquests sistemes és que la deslocalització geogràfica ens limita l'opció de l'accés físic als equips. I també el que un altre administrador pugui obtenir drets d'accés "no lícits". Per això valorarem sistemes d'autenticació més fiables que els simples id's numèrics dels sistemes POSIX[5].

Ara que sabem que les necessitats d'una empresa varien segons el tipus d'aplicació també hem de planificar el pla de contingència (DR) i el pla de continuïtat de negoci (BCP) per a cada un d'aquests emmagatzemaments. Si som capaços de disposar d'un sistema unificat que serveixi tots els protocols abans esmentats i que disposi de mecanismes de replicació i de redundància podem simplificar el pla de contingència.
Si pensem en el pla de continuïtat podem definir que un sistema redundat i dispers en diferents nodes ens permetrà simplificar la seva disponibilitat. La caiguda d'un node no ha de suposar una aturada de servei, el seus propis mecanismes interns o d'implementació de redundància i replicació estan per a evitar-ho. La prevenció davant d'incidents físics a nivell de centre de dades també pot quedar solventat amb aquesta distribució de nodes.

5 POSIX (http://en.wikipedia.org/wiki/Posix)

Gluster

Gluster és una companyia que fou comprada per RedHat l'octubre de 2011. RedHat no ha tancat el desenvolupament Open Source de Gluster, i a més patrocina el seu desenvolupament esperant que la inversió li sigui revertida amb el producte. Ha creat una línia del seu sistema (RedHat Enterprise Storage Server) dedicada a servidors on Gluster és la base de la solució.

Els límits d'un clúster Gluster provenen de la implementació del sistema de fitxers que allotgen els bricks, o nodes de dades. Gluster recomana XFS com a base, per tant està en la suma de la grandària màxima del sistema de fitxers XFS multiplicat pel nombre màxim de nodes que pot gestionar. Teòricament, segons els límits publicats per Gluster: 2^32 * 18 Exabytes són uns, aproximadament 72 Brontobytes[6]. És clar que aquesta dada és merament teòrica, ja que la implementació real és impossible de portar-la a terme. També estan treballant amb la implementació de ZFS i el seu híbrid Open Source BTRFS.

Gluster no disposa d'un director o node gestor. L'accés és concurrent als diferents nodes que formen el pool (el pool són tots els nodes que formen part del clúster). La intel·ligència està a tots els nodes. La caiguda d'un node no posa en perill la integritat del clúster.

Gluster encara no disposa de cap integració amb cap sistema d'autenticació integral com Kerberos però hi estan treballant. Des de la versió 3.5 suporta ACLs sobre NFSv3. Això ens permetria definir les llistes d'accés a través dels clients FUSE, SAMBA i NFS i complir uns dels requisits primordials a nivell de NFS.

Una nova *feature* que implementa són els volums WORM (Write Once, Read Many), tot i que està encara bastant "verda". Aquests tipus de funcionalitats són molt necessàries per a poder complir lleis de protecció de dades. Bàsicament és tracta de definir una política de seguretat que impedirà que qualsevol usuari o administrador pugui esborrar la informació que en aquell volum hi hagi, mentre que s'hi podrà afegir informació.

Gluster disposa de snapshot's (una instantània en un punt de temps) però només per als volums integrats amb el *Block Device translator*. Aquests dispositius són utilitats per sistemes com Cinder (dins el paquet d'OpenStack) per a emmagatzemar imatges de discs virtuals. Però encara no disposen de la opció de fer una snap d'un volum d'arxius normal. Aquesta funcionalitat està prevista per a la versió 3.6.

La gestió d'objectes també és una funcionalitat que està prevista per a la nova versió 3.6. A les versions actuals (la 3.5 en desenvolupament i la 3.4.3 considerada per a entorns productius) caldrà fer aquesta integració mitjançant Swift[7].

6 Brontobyte (http://www.esacademic.com/dic.nsf/eswiki/190284)
7 Swift (https://wiki.openstack.org/wiki/Swift)

Ceph

Ceph és una plataforma de fitxers distribuïts fins fa poc desplegada per la companyia Inktank que, recentment, ha estat comprada per RedHat, que la implementarà en el seu port foli de productes. Tot i que encara no està clar com s'integrarà podria ser que passés a formar part de la seva implementació d'OpenStack o bé del seu RedHat Enterprise Storage Server.

El seu producte permet accés a bloc, a sistema de fitxers i a objectes i permet la realització de snapshots.

Ceph, al igual que Gluster, basa els seus bricks en el sistema de fitxers XFS (encara que suporta ext4 i estan treballant amb BTRFS), per tant els seus límits tornarà a ser el límit de bricks que és capaç de gestionar. Al CERN[8] tenen diverses publicacions on expliquen la implantació que han portat a terme d'un clúster Ceph amb 1024 OSD's i aproximadament 3 Petabytes en un únic sistema de fitxers per un sistema productiu.

Ceph disposa de tres dimonis on cadascú s'encarrega de les funcionalitats principals: Servir fitxers tipus NAS (cephfs), servir objectes (radosgw) i servir imatges (rados). Per la gent de Inktank ha estat força important la monitorització, per saber allà on es dediquen els recursos de maquinari en tot moment. Aquesta monitorització ens serà de vital ajuda per a poder identificar colls d'ampolla o possibles problemes.
Ceph també implementa el framework de Hadoop fent-se compatible amb MapReduce i el sistema de fitxers Hadoop Distributed Filesystem. Sense entrar gaire en detalls comentaré que Hadoop és un sistema de fitxers distribuït molt orientat a entorn de HPC, on les dades queden "compactades" per a ser utilitzades entre diferents nodes.

CephX és una capa d'autenticació que permet ser distribuïda entre tots els nodes directors del clúster. Tot i no ser un sistema estàndard és una aproximació al que seria una integració amb un sistema de Kerberos. Hem de tenir en compte que les capes de segurització que puguem implementar en qualsevol sistema penalitzaran el rendiment. Si la seguretat no és primordial en el nostre entorn cal pensar si aquesta pèrdua de rendiment és assumible o no.

Per a la ubicació i obtenció dels fitxers en els volums o bricks Ceph disposa d'una eina molt interessant anomenada CRUSH. Aquesta eina és capaç de generar un mapa dels nodes i determinar segons uns criteris de proximitat, de latència o de redundància on s'ha d'ubicar aquesta informació, o a quin node cal que accedeixi un client. Trobo molt interessant que permet definir una política personalitzada en funció dels nostres criteris personals, de tal forma que puguem intervenir en la decisió de l'elecció del node. Tanmateix aquest eina ens pot ajudar a determinar on tenim els problemes al integrar-se amb les eines de monitorització.

Ceph també disposa d'una *feature* molt interessant. És el *tiering*, o el que seria ubicar la informació en diferents tipus de discos segons el seus accessos a discos de més o menys rendiment. El que ens permet aquesta opció és obtenir un rendiment màxim per a les dades d'accés més freqüents, i no haver de dedicar discos d'altes prestacions (i per tant més cars) a dades que no són tan consultades. Les dades que estan "vives", o que tenen accessos més recents, pugen de nivell i passen als discos més ràpids.

8 CERN (http://home.web.cern.ch/)

En configuracions idònies podem disposar d'un nombre reduit de discos SSD per a les dades accedides a les últimes hores, discos SAS de 15K RPMs per a les dades accedides en els últims dies i la resta de la informació romandria a discos NL-SAS (Near Line SAS) o SATA d'alta capacitat i baix rendiment.

Xtreem

XtreemOS és un sistema operatiu que de base ja està preparat per treballar com si fos un clúster de servidors. Implementen el XtreemFS com la seva solució per a tenir un sistema de fitxers distribuït. Aquesta pot ser extreta del seu sistema operatiu i instal·lada sobre qualsevol GNU/Linux.

Directament no suporta l'accés a bloc però sempre es pot fer mitjançant la inclusió d'una passarel·la entre el clúster Xtreem i un sistema que serveixi a bloc mitjançant qualsevol implementació iSCSI Target.

Xtreem es basa en tenir un servei director (DIR). Aquest servei el fa servir el servei de metadades i replicació (MRC) que posa en contacte el client amb els bricks (OSD) que emmagatzemen la informació. Hi ha un servei de replicació (RMS) que és el responsable de decidir quan una rèplica cal destruir-la o crear-la, i quina política ajusta millor per a la decisió de sobre quins bricks cal ubicar-la. Aquesta política es pot definir dins d'una sèrie de polítiques predefinides d'entre una política aleatòria, per proximitat segons el rang d'IPs dels OSDs o per el nom DNS, per zones.

Permeten la creació de snapshots asíncrones però el seu sistema és molt poc tolerant a altes latències i amples de banda reduïts.

L'accés des d'estacions Windows es pot fer mitjançant una passarel·la sobre SAMBA o bé amb un client XtreemFS per a Windows. El desplegament sobre parcs informàtics grans seria un problema que caldria contemplar a banda.

De cara als clients Unix/Linux, XtreemFS compleix els estàndards de NFSv3 amb suport de llistes d'accés. Permet la integració amb certificats digitals per a l'autenticació. Sobre la seva plataforma XtreemOS i el seu producte Virtual Organization tanquen el cercle de la seguretat, facilitant-ne la integració i la gestió.

Plasma

Plasma està patrocinat per CamlCity. Un dels projectes que porta en marxa des de 2010 és un sistema de fitxers distribuït anomenat PlasmaFS. Aquest implementa les funcionalitats de MapReduce[9].

Plasma està molt enfocat per a entorns web. Les comunicacions entre les passarel·les i els nodes es realitza per comunicacions xifrades. Aquest punt que ofereix una capa de seguretat extra, també penalitza el rendiment. El xifrat de les comunicacions requereix de consum de processador i d'ample de banda.

9 MapReduce (http://research.google.com/archive/mapreduce.html)

Disposa de funcionalitats avançades de selecció del node, però no és capaç de realitzar snapshots dels volums.

A trets generals sembla un projecte que encara ha de madurar molt i no crec que estigui preparat per a entorns productius reals.

OpenAFS

OpenAFS és la evolució de l'Andrew FileSystem, desenvolupat a la universitat de Carnegie-Mellon. És un sistema força estès, sobre tot en àmbits universitaris. És un sistema de provada resistència i creixement. No és un sistema que estigui pensat per a entorns WAN.

OpenAFS ens proporciona un espai de noms d'entrada única. Això vol dir que l'usuari disposarà d'un punt de muntatge, i per a ell, tot semblarà que pertanyi al mateix sistema de fitxers. Cada carpeta o directori pot formar part d'un altre node o clúster, disposant de delegació per cel·les.

Sembla que últimament el projecte d'OpenAFS està tornant a estar actiu, però havia estat un temps aturat en el seu desenvolupament.

OpenAFS és un sistema pensat per a entorns distribuïts de bon principi i, com a tal, porta un sistema d'autenticació de Kerberos de desenvolupament intern, tot i que fan molts esforços per a integrar-se amb d'altres externs.

Un clúster d'OpenAFS s'organitza en cel·les, que alhora estan formades per servidors. Cada servidor executa una sèrie de processos en funció dels rols que n'és responsable. Així hi podem trobar:

- File-server que s'encarrega d'emmagatzemar la informació al disc local i de transferir-lo de/a les estacions clients.
- Basic OverSeer Server (BOS) que s'encarrega de supervisar el correcte funcionament dels serveis que corren al propi servidor.
- Protection Server que gestiona quins usuaris tenen accés als fitxers.
- Volume Server que s'encarrega de gestionar totes les accions sobre els volums.
- Volume Location Server que manté la base de dades de localització de volums en els diferents nodes.
- Update Server que s'encarrega de mantenir les versions dels binaris del serveis actualitzats.
- Backup Server que s'encarrega de mantenir la base de dades de backup.
- Cache Manager, és un component que resideix a les estacions clients.

OpenAFS disposa d'un client per a Windows que ens permet parametritzar perfectament l'accés als seus clústers.

OpenAFS disposa de funcionalitats per a fer snapshots a nivell de cel·la, però no està indicat per a allotjar volums de bloc.

Taula comparativa

Figura 3: Taula comparativa

	Llenguatge	Kernel Patch	FUSE Compliant	SPOF(1)	Entorns Productius	SNAPs	iSCSI	FCoE	Replicació
Gluster	C++	No	Si	No	Si	No(2)	Si	Si	Si
XtreemFS	Java	No	Si	Si	Si	Si	No	No	Si
PlasmaFS	Python	No	No	Si	No	No	No	No	Si
OpenAFS	C++	Si	Si	No	Si	Si	No	No	Si
Ceph	C++	Si	Si	No	Si	Si	Si	Si	Si

(1) SPOF: Single Point of failure, o en Català, Punt únic de fallida significa que el sistema té un punt crític central. Aquest punt sovint es pot redundar mitjançant sistemes d'alta disponibilitat, però són sistemes actius-passius. Aquest punt significa que hi ha algun punt del sistema que pot posar en risc l'accés al servei. Un sistema director, o arrel de la cel·la són els punts calents
(2) En el *roadmap* per a la versió 3.6.

Pilot

Descripció del pilot

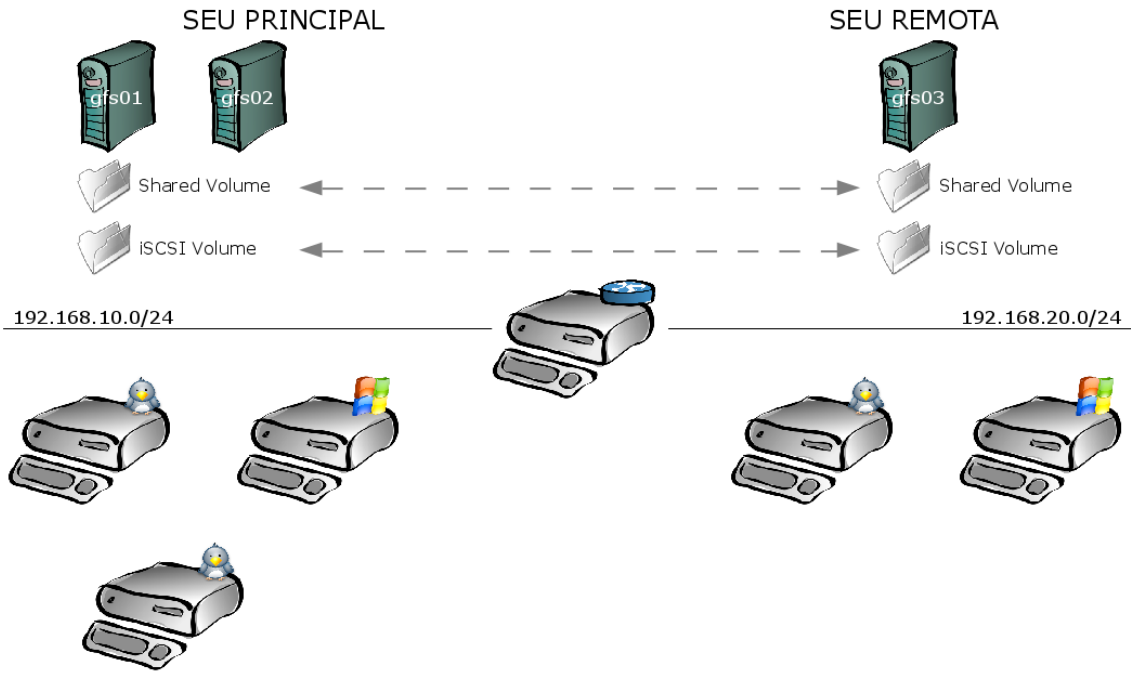
Un cop vistes les necessitats que tenim hem de posar en marxa una proba de concepte, un petit pilot on validarem el producte. De les diferents opcions que hem vist sembla que tant Gluster con Ceph estan com a punta de llança en els SDS. Gluster és una de les bases més sòlides per a la construcció d'un IaaS (Infraestructure as a Service) amb OpenStack, i la d'una solució de PaaS (Platform as a Service) com OpenShift. Ceph està ara mateix en un moment d'incertesa degut a la compra per part de RedHat i caldrà veure cap a on evolucionarà el seu producte, ja que, en gran mida comparteix característiques amb Gluster. A més, Ceph requereix d'una infraestructura més àmplia per a muntar el pilot.

D'altra banda veiem que Gluster és un producte consolidat. La seva comunitat cada dia és més productiva i el seu producte evoluciona ràpidament escoltant les necessitats dels usuaris. RedHat, i també CentOS i Fedora, l'han adoptat i també està focalitzant molt els seus esforços per a aconseguir implantar solucions empresarials al món real. Les funcionalitats descrites a les característiques de la web de Gluster fan que sigui una opció molt prometedora, i, a priori, sembla relativament senzilla de posar en marxa.

Així cal validar que una de les solucions és o pot ser un clúster de Gluster. Per tal de posar-lo en pràctica necessitarem muntar una estructura bàsica que veurem detallada a continuació i que haurà de respondre a les necessitats que ens hem marcat.

L'esquema global del sistema (Figura 4) tractarà dues seus, una principal i una remota. Un sistema intermedi ens simularà la connectivitat d'Internet i ens permetrà el control per a simular la qualitat d'una línia pública.

Figura 4: Esquema general de xarxa



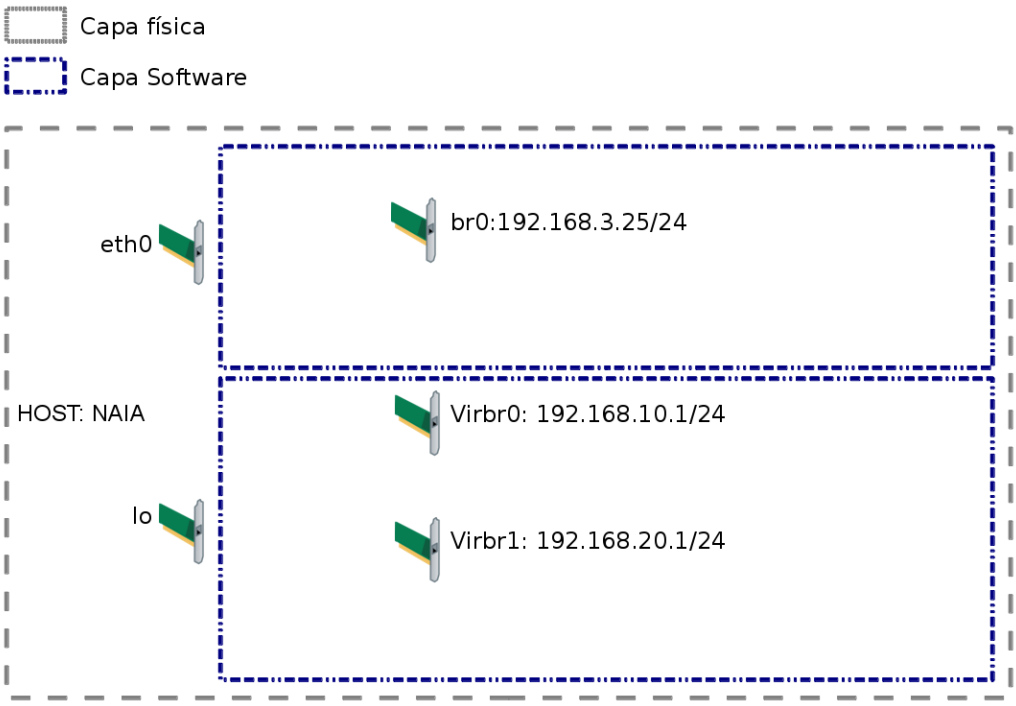
Networking

L'esquema de xarxa vindrà determinat per la configuració física de l'entorn. Per a simular aquesta topologia utilitzaré les eines de virtualització virsh i kvm.

La figura 5 ens mostra l'esquema de xarxa format per dues interfícies virtuals (virbr0 i virbr1) que representen les xarxes de diferents ubicacions físiques del negoci. La primera representarà la seu principal i la segona una oficina remota ubicada a la Xina, per mostrar la pitjor de les situacions (worst case scenario). Aquestes estan vinculades a la interfície física "loopback" per tal d'eliminar el coll d'ampolla[10] que suposa la xarxa d'un GE. Els clients hi accedeixen mitjançant la interfície virtual br0 sobre una xarxa física (eth0), així que segur que tenim el límit del GB.

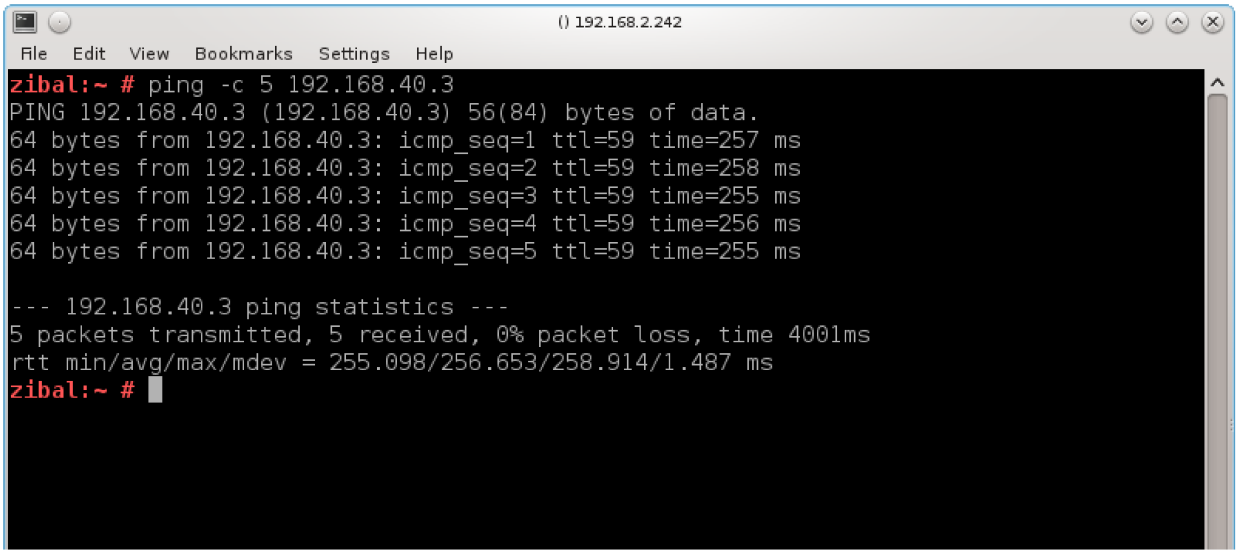
10 Bottleneck (http://en.wikipedia.org/wiki/Bottleneck#Traffic)

Figura 5: Esquema xarxa lògica-física host



Els servidors del primer clúster (gfs01 i gfs02) estan vinculats a la interfície virbr0. El tercer node (gfs03) el tindrem a la unitat de negoci externa. Les comunicacions entre les unitats de negoci es faran mitjançant una connexió WAN on la seu remota disposa d'una connexió de 20Mbps síncrona. Com disposar d'un sistema remot on efectuar el pilot ha estat impossible, simularé les comunicacions que tenim actualment amb una oficina de la Xina (Shanghai). Les latències reals entre aquestes dues xarxes és:

Figura 6: Latència real comunicacions Xina

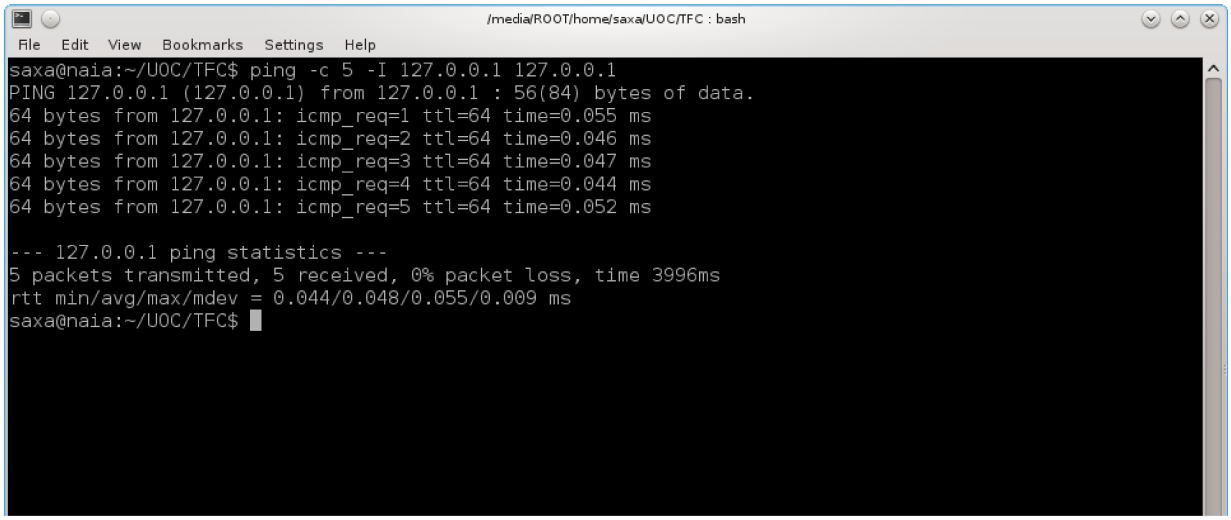


Primer observem quin rendiment ens ofereix el sistema sense modificar ni alterar cap tipus de comportament de la latència ni de l'ample de banda:

Per la comunicació que hi haurà a través de la interfície *loopback*, és a dir, la comunicació que hi haurà entre els diferents nodes de cada seu, primer a nivell físic.

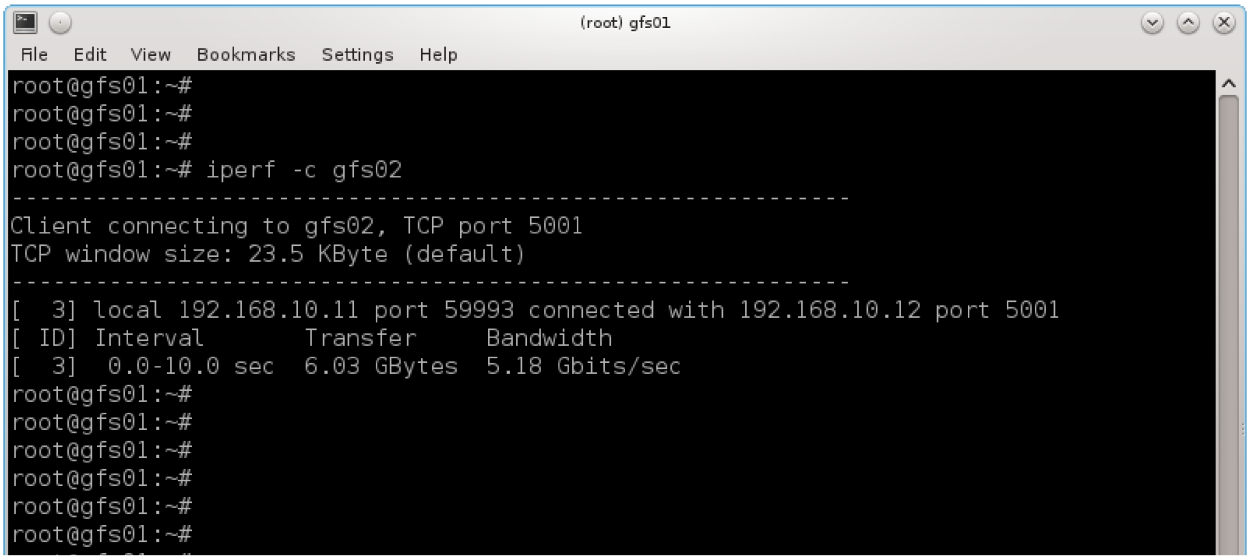
La figura 7 ens mostra la latència mesurada amb un "ping".

Figura 7: Latència interfície loopback



L'ample de banda el mesurarem amb la eina "iperf" a la figura 8:

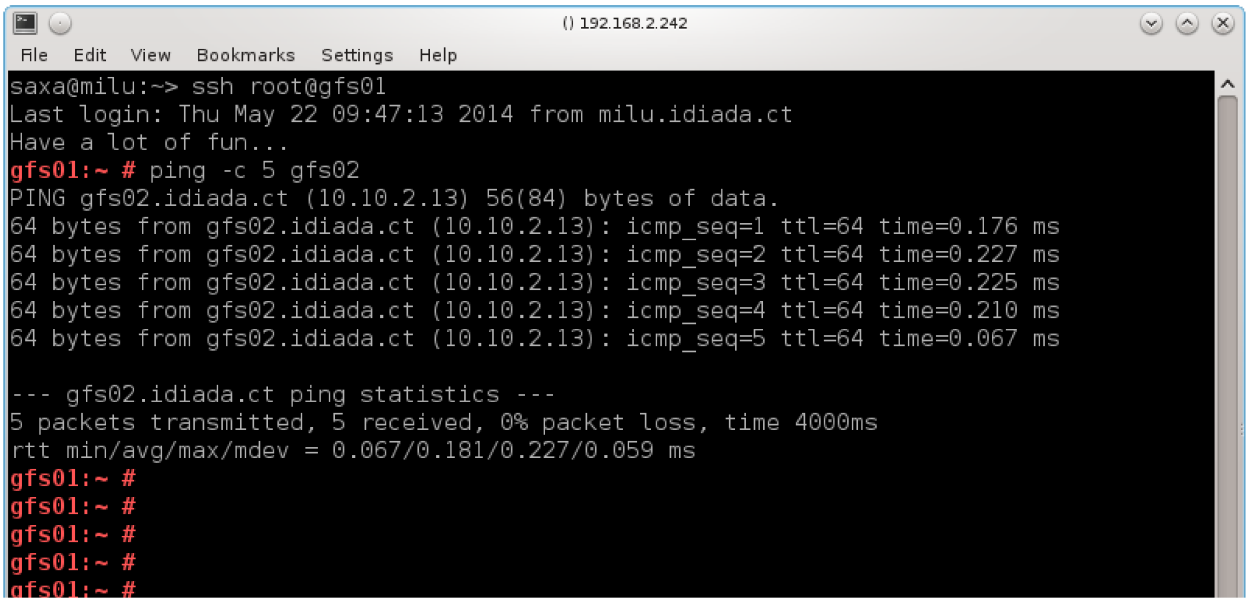
Figura 8: Ample de banda interfície local



També podem comprovar que les diferents capes de virtualització ens penalitzen el rendiment i cal tenir-lo en compte:

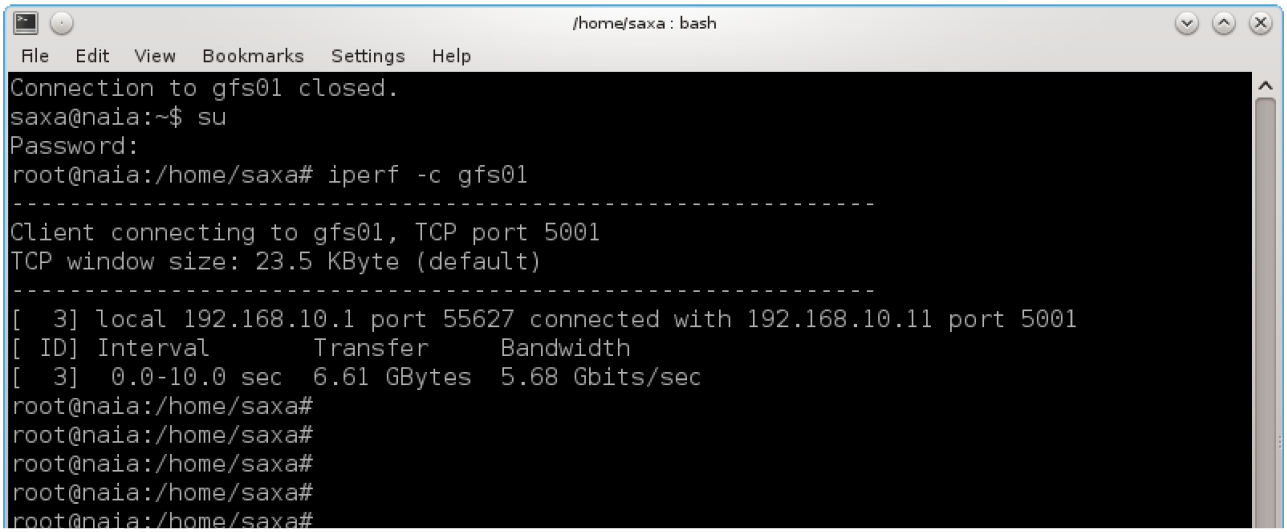
La latència entre nodes virtualitzats amb un simple “ping”:

Figura 9: Latència intra-node sense modificar



I l'ample de banda:

Figura 10: Ample de banda màxim entre nodes virtualitzats



Per tal de simular les mateixes característiques de comunicacions a l'entorn simulat utilitzarem les eines Advanced Routing & Traffic Control incorporades al nucli del host principal.

Per al nostre cas particular executarem les instruccions:

```
root@naia:~# tc qdisc add dev vnet2 root handle 1:0  cbq bandwidth 20mbit avpkt 100 cell
8
```

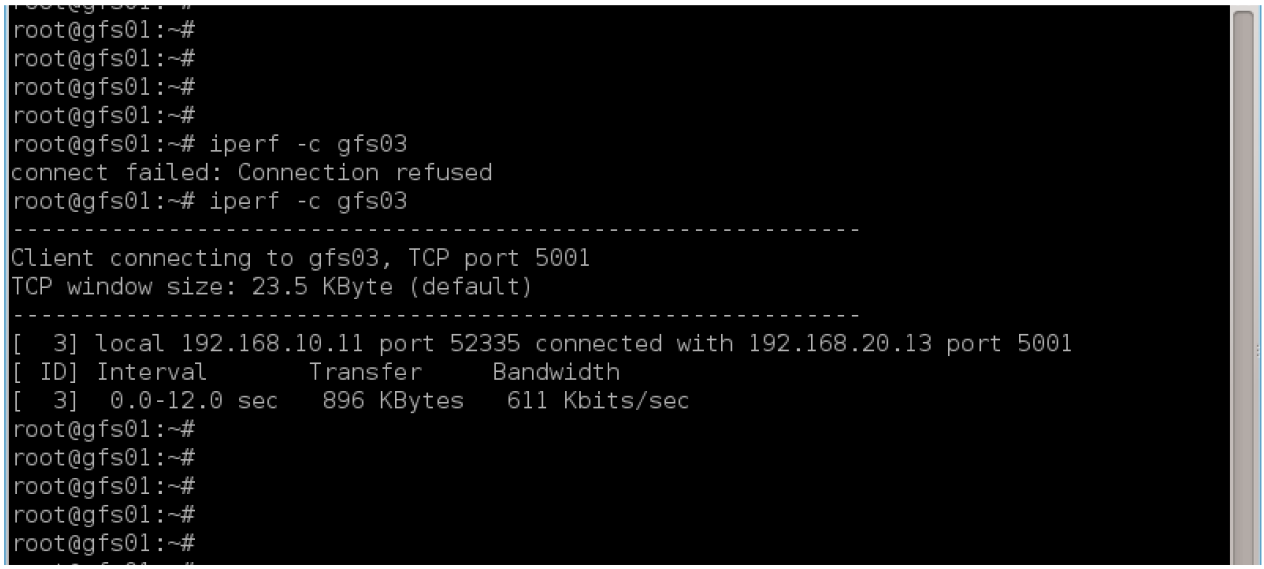
Que defineix un ample de banda màxim de 20Mbits.

```
root@naia:~# tc qdisc add dev vnet2 parent 1: netem delay 260ms 20ms loss 1% reorder 5%
50%
```

Que afegeix una latència de 260ms amb una variabilitat de 20ms, una pèrdua d'un punt percentual i una reordenació del 5%.

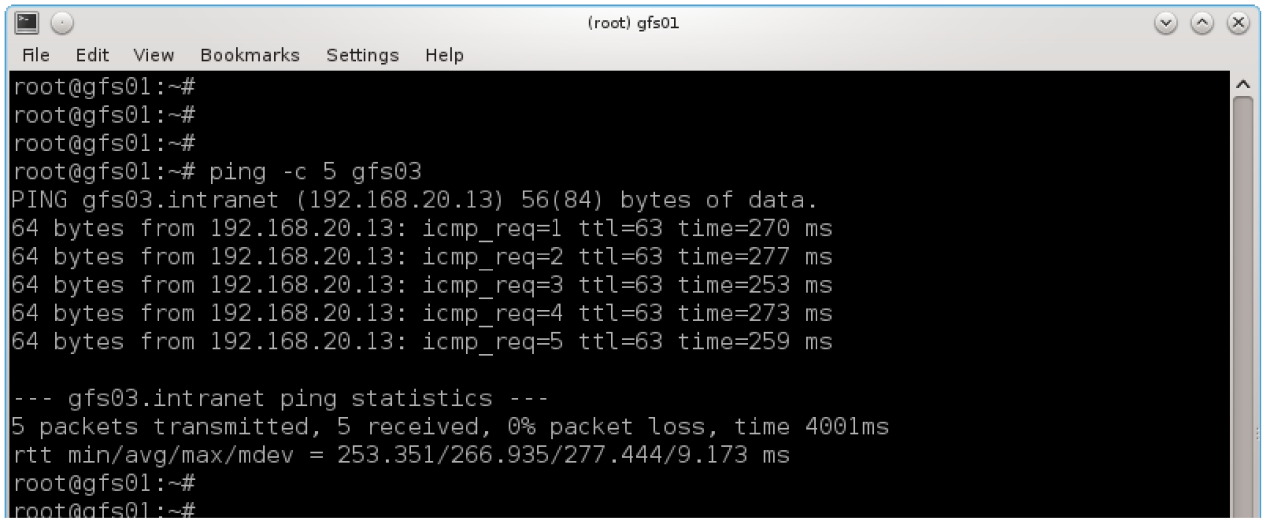
Podem apreciar el canvi de comportament un cop he aplicat aquestes instruccions. A la figura 8 amb el "iperf" on podem apreciar la baixada de rendiment:

Figura 11: Ample de banda intra-node un cop restringit



I la latència:

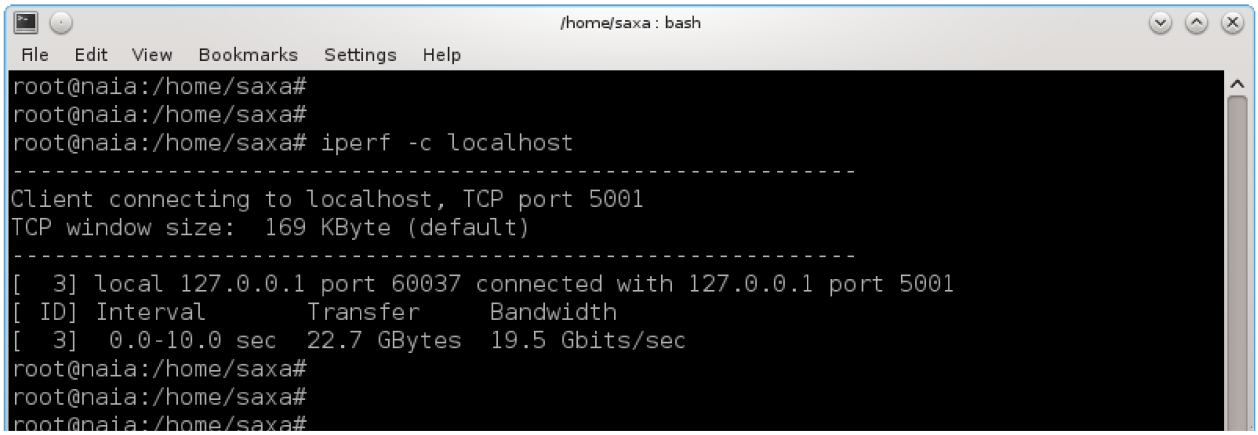
Figura 12: Latència intra-node modificada



Així doncs també és important saber quin rendiment podem obtenir de la comunicació per xarxa:

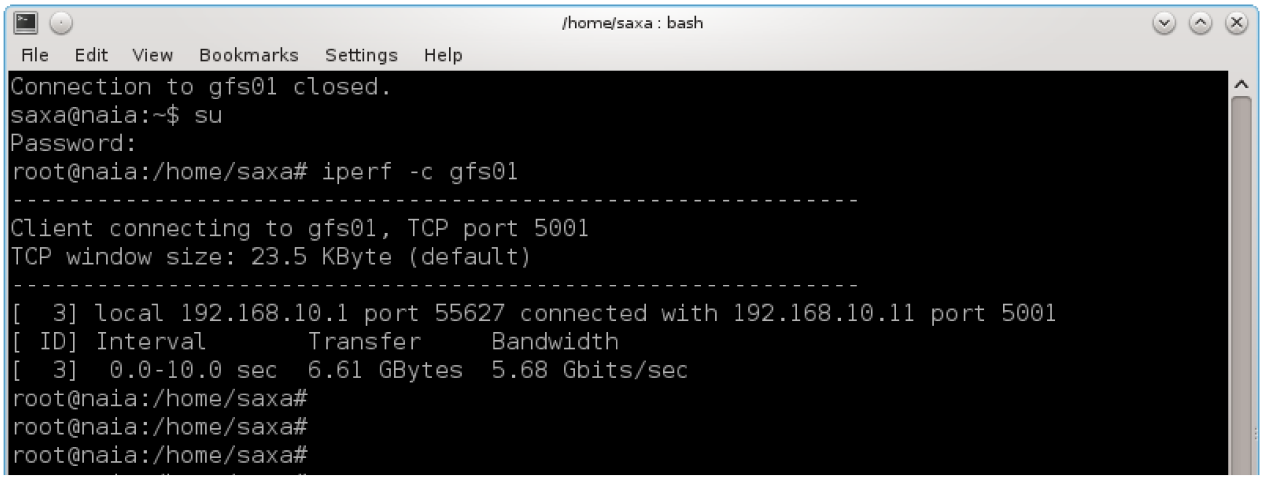
Mirem l'ample de banda màxim que podem obtenir de la interfície local.

Figura 13: Ample de banda sobre el loopback



El rendiment que podem esperar contra l'equip virtualitzat.

Figura 14: Ample de banda sobre un node virtual



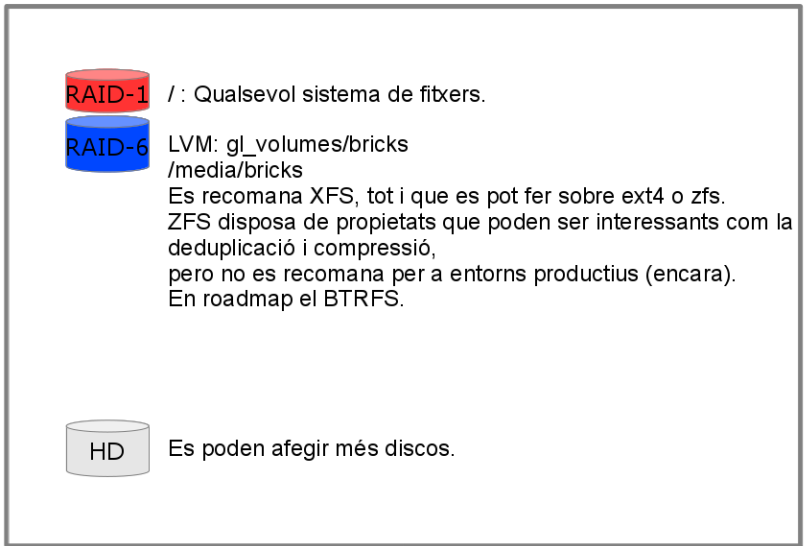
Servidors

Per al pilot vaig començar amb una instal·lació sobre Debian 7 (Wheezy). Després de moltes probes i configuracions Debian no té disponible el "Thin provisioning", que és una característica imprescindible per a realitzar snapshots i la definició del nou *Block Device Translator* (BD-Xlator) de base que promou Gluster. Així que finalment vaig optar per canviar a Fedora 20, la última versió i que segurament s'assembla més a RedHat.

Els nodes servidors de Gluster físicament es poden muntar o pensar de dues formes, cadascuna de les quals te els seus pros i cons:

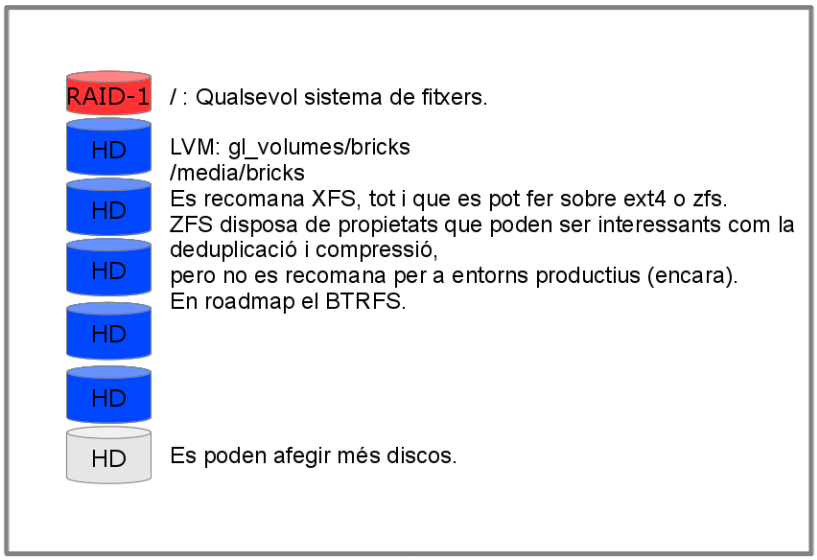
Amb un maquinari o controladora de disc que permeti la construcció de RAID's i que ens cobreixi la possible fallada d'un disc dur.

Figura 15: Node amb maquinari dedicat a la integritat dels discos



O bé, podem disposar de discos simples i definir els nostres volums de replicació, de tal forma que sigui Gluster qui s'encarregui de la redundància de les dades. Aquesta configuració és la recomanada per RedHat en el seu RedHat Storage Server 2.0 i la tendència que segueixen tots els productes SDS. A més centralitzem tota la gestió de la integritat del maquinari en un únic punt, cosa que ens evitarà administració diversificada havent d'instal·lar controladors específics per a cada controladora RAID de discos.

Figura 16: Node sense maquinari dedicat a la integritat dels discos



Disposar d'un sistema de RAID facilita la gestió dels nodes i simplifica la gestió dels volums, però cal tenir en compte que els grups de replicació seran força més grans al fer-ho amb volums (arrays) més grans. També provoquem una dispersió de la integritat de les dades, delegant-la a la controladora de discos i no permetent que la gestioni el propi Gluster.

Per al pilot simplificaré la instal·lació i disposarem de cinc discos virtuals:

vda (8GB): Disc d'arranc per al sistema operatiu.
vdb (4GB): Disc per al brick on crearem un volum llògic(vdb).
vdc (4GB): Disc per al brick on crearem un volum llògic(vdc).
vdd (4GB): Disc per al brick on crearem un volum llògic(vdd).
vde (4GB): Disc per al brick on crearem un volum llògic(vde).

Com a sistema de fitxers de *facto* he triat XFS, tot i que BTRFS i ZFS estan disponibles, per a aquest pilot, els tamanys del pilot no superen les limitacions de les versions bàsiques.

Cada node disposarà de dues interfícies virtuals, una per a la comunicació cap els clients i una altra per a la comunicació intra-node. Aquest punt cal especificar que és molt important disposar d'una comunicació privada entre nodes per tal d'evitar que l'ús intensiu d'un client pugui afectar el rendiment global del clúster.

Cada node disposarà d'un processador i de 1G de RAM.

Com podem veure aquesta configuració no és especialment potent però per al pilot en tindrem suficient.

A la primera fase del pilot he preparat només dos nodes. Aquests estaran ubicats a la seu principal, on hi configuraré dos volums Gluster:

- Un volum replicat, que vindria a semblar un RAID-1 entre dos equips.

- Un volum *stripped*, que vindria a semblar un RAID-0 entre dos equips.
- Veurem que existeix el concepte de volum distribuït, que equivaldria a un sistema JBOD (Just a bounch of disks). Aquest tipus de configuració afavoreix el creixement i la disponibilitat, degut a que la pèrdua d'un node no implica la pèrdua de la informació, només la part que hi ha sobre el node fallat.

A la segona fase hi incorporaré el node de la seu remota (Xina) i estendré aquests volums per a que hi hagi dos tipus de replicació:

- Replicació síncrona: El node gfs03 formarà part del mateix clúster i rebrà les peticions com si formés part de la xarxa de la seu principal.
- Replicació asíncrona: El node gfs03 rebrà les sincronitzacions efectuades al volum del clúster principal. El volum quedarà en mode de consulta al node remot.

Clients

Com a clients tindrem un client Linux (Debian Wheezy) que podrà fer servir el client natiu de Gluster (Filesystem in User Space o FUSE) o bé el client de NFS. L'altre client serà un Windows7 que hi accedirà mitjançant el SAMBA muntat en alta disponibilitat al clúster de Gluster.

Tal i com hem definit a la part d'Anàlisis de necessitats, ja hem vist que necessitem tres tipus de sistemes d'accés: Fitxer, bloc i objecte.

Pel que fa a fitxer tenim dos protocols (CIFS i NFS o Gluster) per cobrir tot el ventall de clients que necessitem (Windows, Unix, Linux, AIX, etc).

Els clients Windows els podem cobrir mitjançant la instal·lació d'un client NFS a cada estació (cosa que pot tenir un cost força elevat) o mitjançant la instal·lació dels serveis de SAMBA (Servidor Common Internet File System - CIFS) al clúster.

SAMBA és un servei que s'instal·la sobre sistemes Unix i Linux i que simula un servidor de fitxers Windows. El seu nom ve de la versió del protocol anterior (Server Message Block, i que posteriorment es va reanomenar com a CIFS). En el moment de la creació del projecte SAMBA, les seves funcionalitats eren força bàsiques, però avui en dia, es tracta d'un projecte estable i molt consolidat que, fins i tot, te estudis de rendiment que mostren servidors de SAMBA oferint més rendiment que un servidor natiu de Windows.

En aquest últim cas també disposem de dues formes de parametritzar aquesta instal·lació.

- SAMBA amb HA (Heartbeat[11])
- SAMBA distribuit (CTDB[12])

La primera opció per a un entorn de probes és suficient, però per a entorns productius semblaria poc eficient. Un client de Windows atacant a un clúster de SAMBA distribuït,

11 Heartbeat (http://www.linux-ha.org/wiki/Heartbeat)
12 CTDB (https://ctdb.samba.org/)

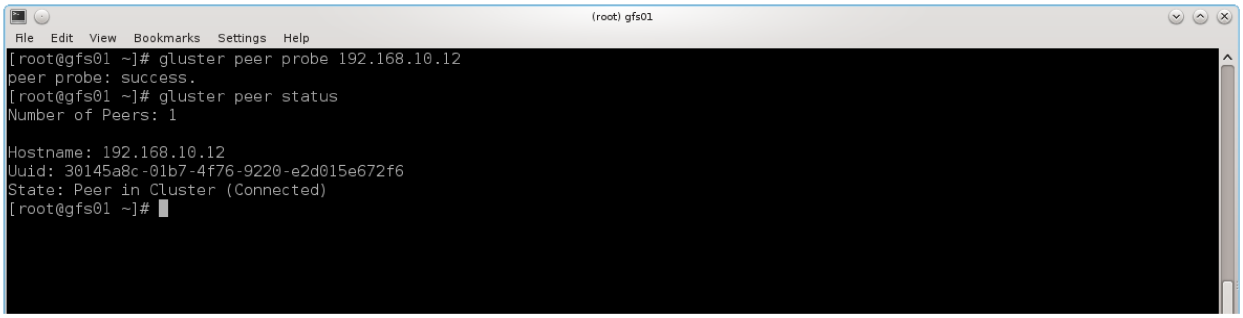
permetria balancejar la càrrega sobre els diferents nodes, ampliant-ne l'ample de banda disponible per a cada client. A més, permetria que un client, tot i la caiguda del node on estigués connectat, no hagués de forçar a una reconnexió. SAMBA va introduir el manteniment d'estat dels fitxers sobre un fitxer de base dades. Si aquest l'allotgem sobre un volum replicat a tots els nodes, totes les instàncies de SAMBA són capaces de conèixer l'estat de les sessions i bloquejos als fitxers. Per a distribuir la càrrega sobre els diferents nodes caldria muntar una delegació de zona per a que el SAMBA indiqui quin és el node més adient per a cada connexió segons uns criteris a definir (proximitat, càrrega, etc).

Per al muntatge actual ho he preparat tot seguint el model de SAMBA amb HA ja que el muntatge amb CTDB és una mica més complexe i per a les probes funcionals no cal.

El principal inconvenient que hi ha en muntar SAMBA amb un heartbeat és que les connexions de CIFS són orientades a sessió i, per tant, si el node que està actiu s'atura, els clients haurien de re-establir la sessió amb el node alternatiu.

Però comencem amb la creació del pool (o clúster) d'una forma molt senzilla. Un cop tenim els paquets, o les fonts compilades, i instal·lades, des d'un dels nodes executarem:

Figura 17: Afegir un node al clúster



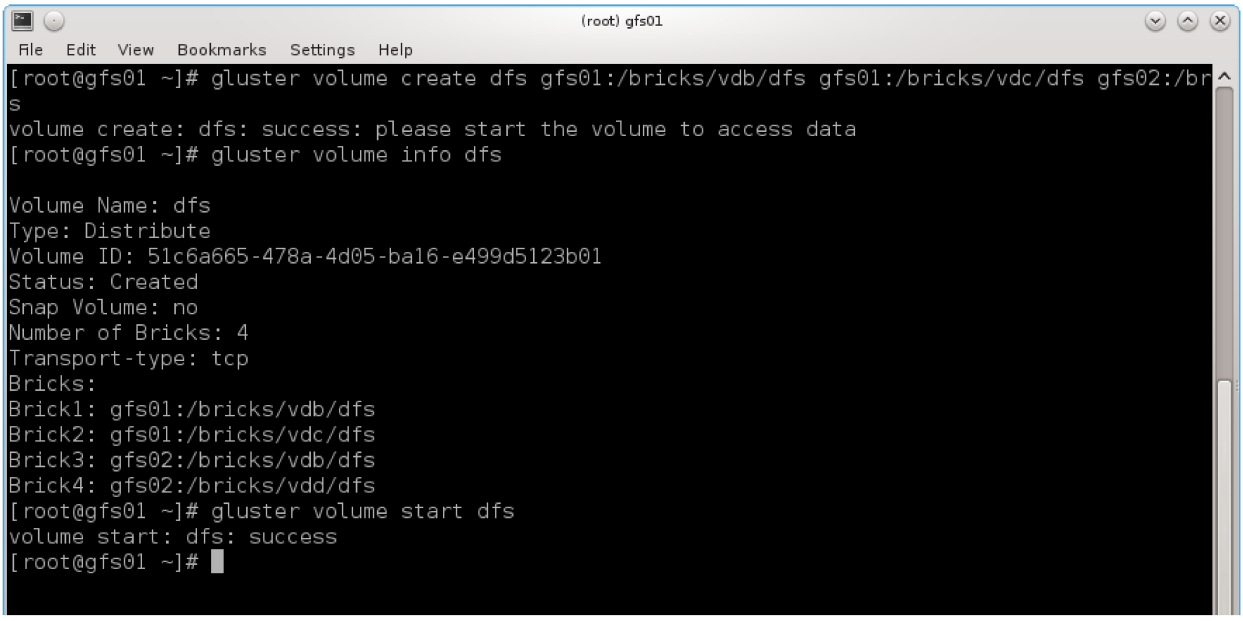
Amb una instrucció tant senzilla, el sistema verifica que hi hagi visibilitat entre ells i es donen d'alta entre ells. Posteriorment podem veure que la instrucció de “peer status” ens desvetlla que es veuen entre ells.

Intentem crear les probes bàsiques per a un primer volum.

Quins tipus de volum tenim disponibles? Les opcions podríem mirar d'equiparar-les a un sistema RAID. Així disposem de distribució (JBOD), stripping (RAID-0), replica (RAID-1) i totes les combinacions que vulguem: stripping + replica (RAID 10), etc.

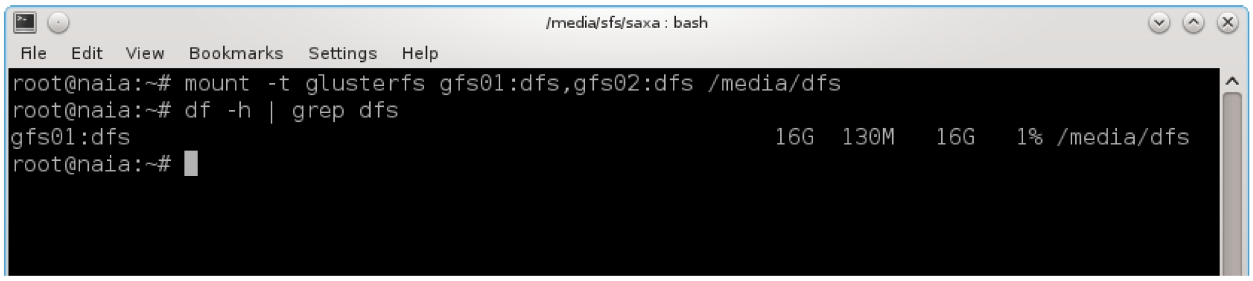
De moment farem tots els tests amb un volum distribuït:

Figura 18: Crear volum DFS amb quatre bricks



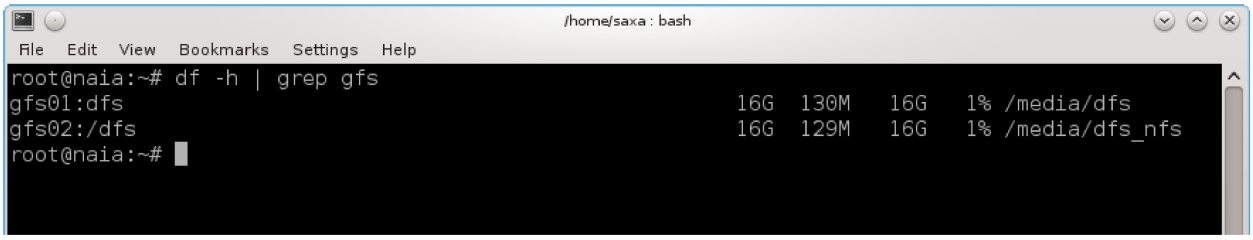
Amb això he creat un volum amb quatre bricks, distribuïts en dos nodes. He mostrat la informació del volum i finalment l'he iniciat. Des d'un client ja el tenim disponible:

Figura 19: Mount específic per FUSE



Quan fem servir el client de FUSE i hi ha una xarxa interna de comunicació és convenient especificar quins són els servidors que formen part del pool.

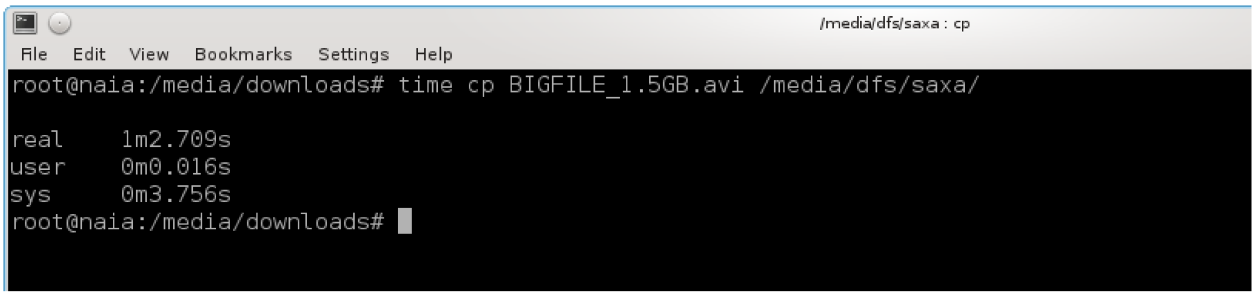
Figura 20: Volum vist des d'un client FUSE i NFS



Ja podem veure el volum muntat al client, per client FUSE i per NFS. A primer cop d'ull podríem pensar que si algun dels nodes fallés, perdríem de vista el volum. En aquest cas perdríem part de la informació ja que només està distribuït.

Veiem si podem afegir-hi algun brick i re-balancejar-lo, però primerament hi haurem de copiar alguna cosa per a poder veure el seu re-balanceig:

Figura 21: Copiar dades al volum DFS



A continuació podem veure com ha distribuït la informació sobre els discos físics:

Figura 22: Distribució física de la informació a GFS01

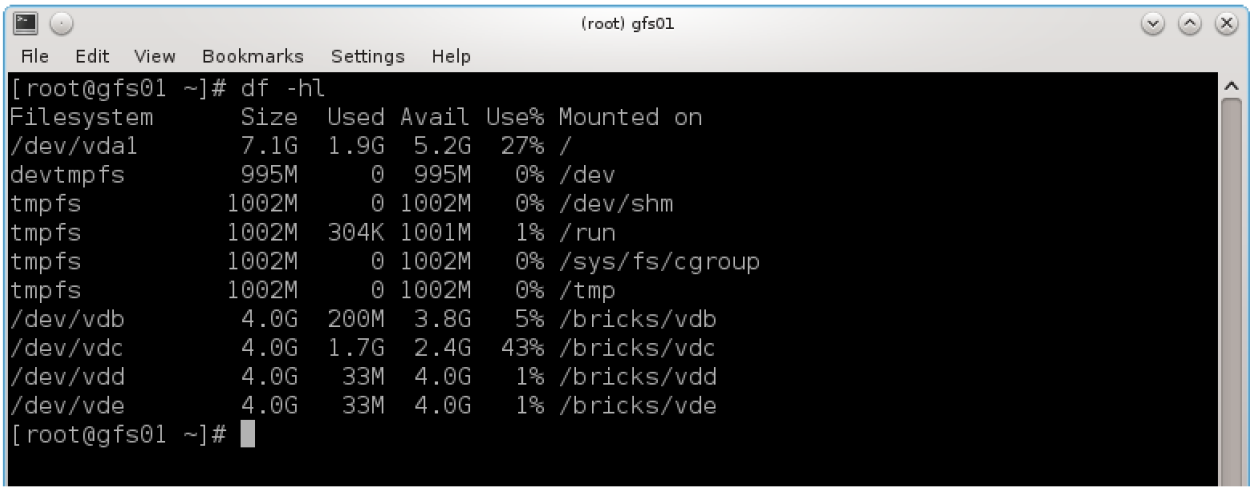
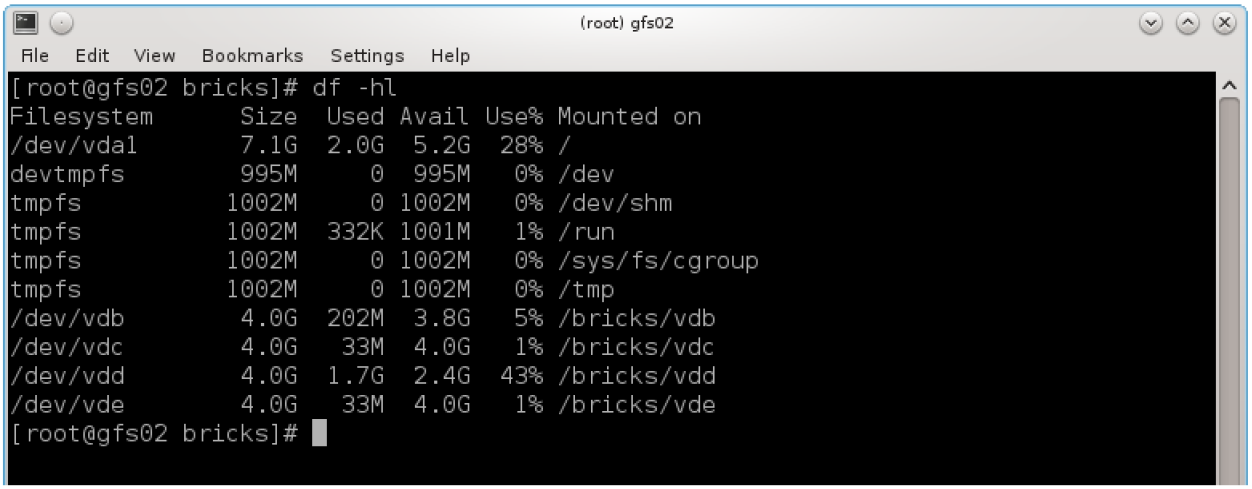


Figura 23: Distribució física de la informació a GFS02

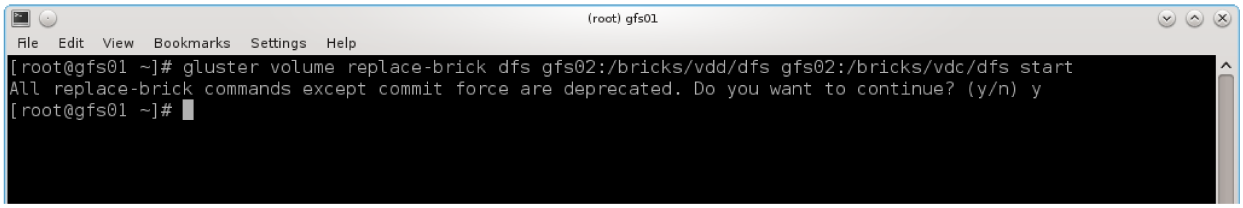


El que podem veure és que la informació no s'ha balancejat equitativament sobre els discos que hi teníem assignats. Podem re-balancejar la informació a veure si canvia aquesta distribució, aquesta operació ha de ser una operació que haurem d'anar executant a mida que anem afegint o descartant nodes i bricks.

Figura 24: Rebalance start

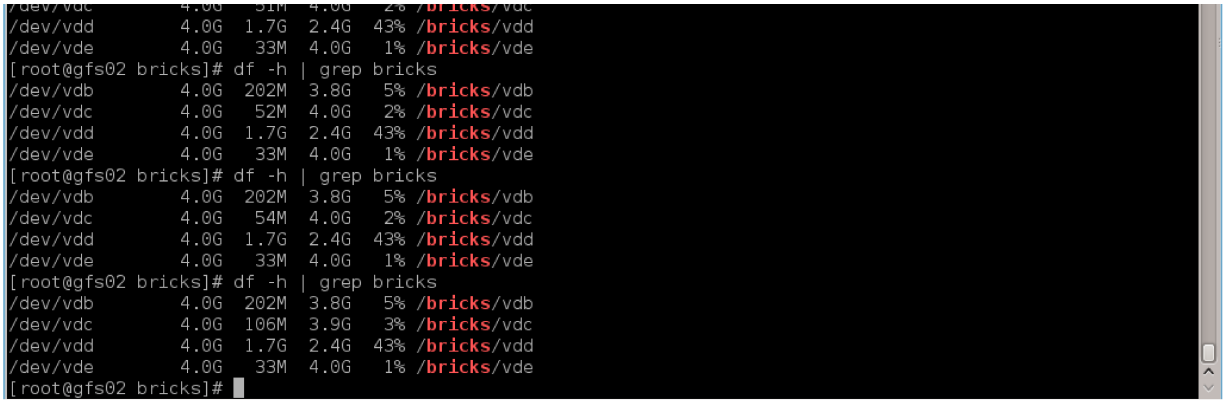
Podem necessitar moure la informació d'un brick a un altre. A l'exemple estem veient com he assignat els volums "vdb" i "vdd". Al primer node he assignat "vdb" i "vdc". Vaig a moure la informació del "vdd" al "vdc".

Figura 25: Replace brick



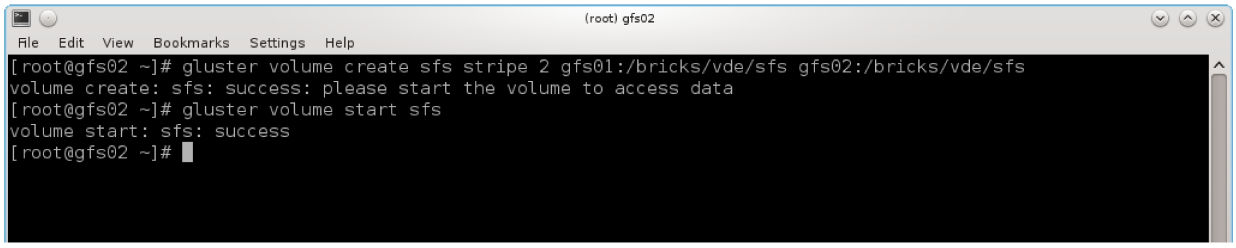
I podem veure com la informació va canviant de brick progressivament.

Figura 26: Replace brick progress



Mirem de crear un altre tipus de volum. Gluster recomana la no utilització dels volums stripped excepte per a casos d'alta càrrega i necessitat de rendiment. Tal i com hem vist a l'anàlisi de necessitats, aquest és un dels escenaris que tindrem.

Figura 27: Crear volum Stripped



Mirem d'executar el set de probes bàsiques:

Figura 28: Descompressió sobre un volum Stripped

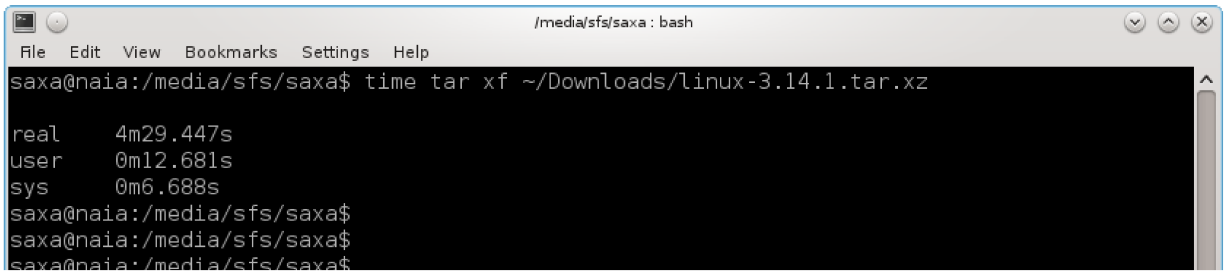
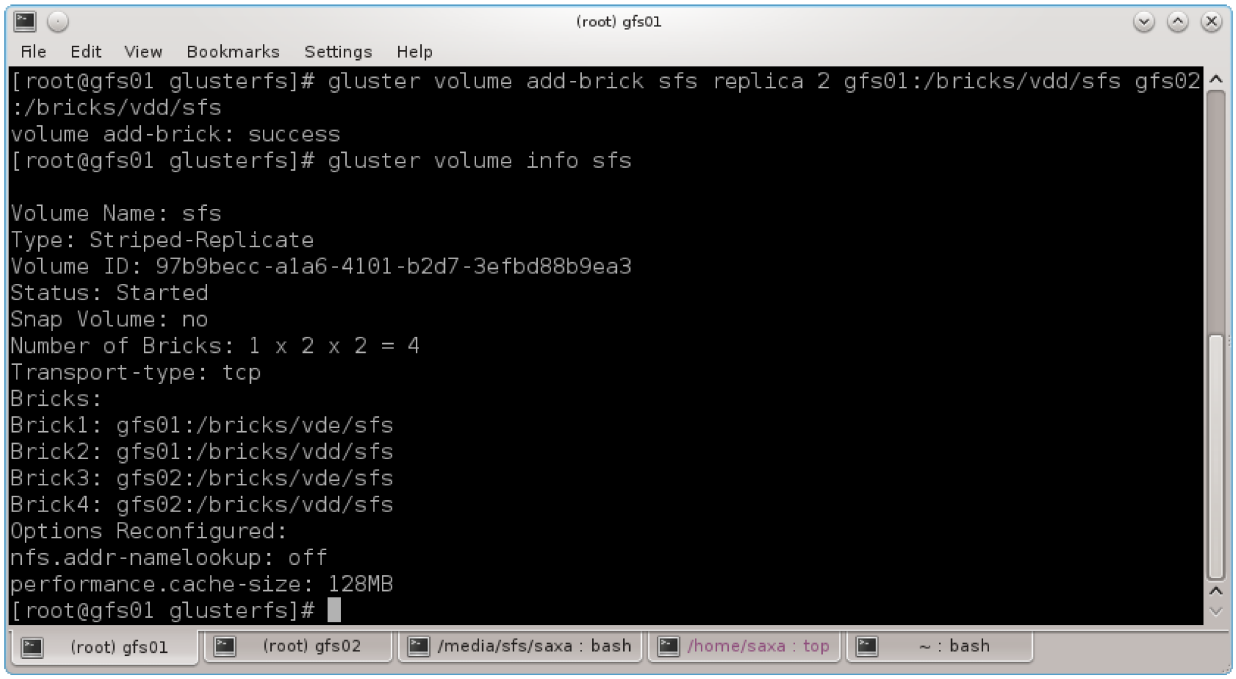


Figura 29: Copiar un únic fitxer gran sobre el volum Stripped

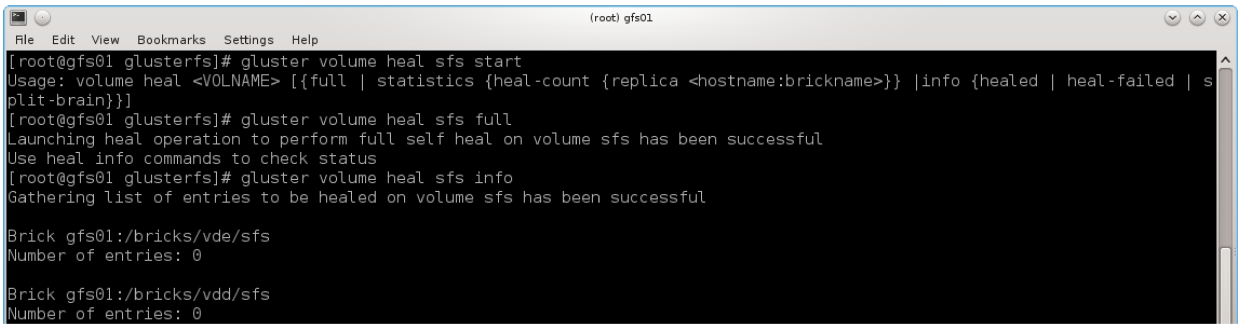
Ara afegirem uns bricks al volum Stripped per a afegir-hi redundància:

Figura 30: Add-bricks sobre SFS



I executem un check del volum:

Figura 31: Check del volum SFS



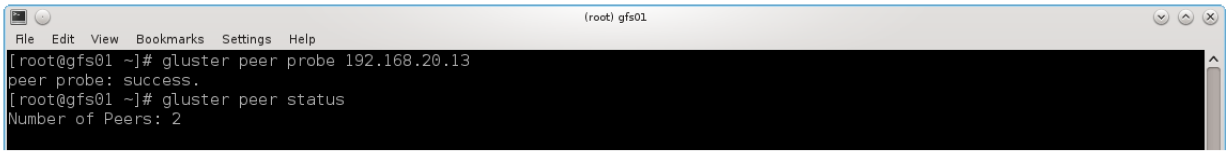
Aquesta operació ens afegirà les rèpliques que li he demanat al sistema, i seria el substitut d'un sistema de RAID-1. El control el tindríem en el mateix Gluster. Podem veure el resultat amb:

Figura 32: Check del volum SFS

Ara, per fi, començaré a fer servir el node remot, i recordem que tenim aplicades unes latències força altes i un ample de banda restringit.

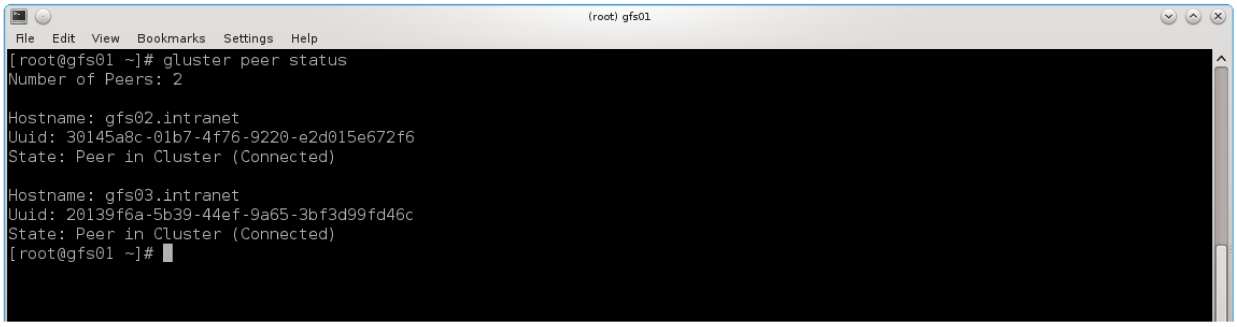
Primerament contactaré amb el node gfs03.

Figura 33: Contactar amb node gfs03



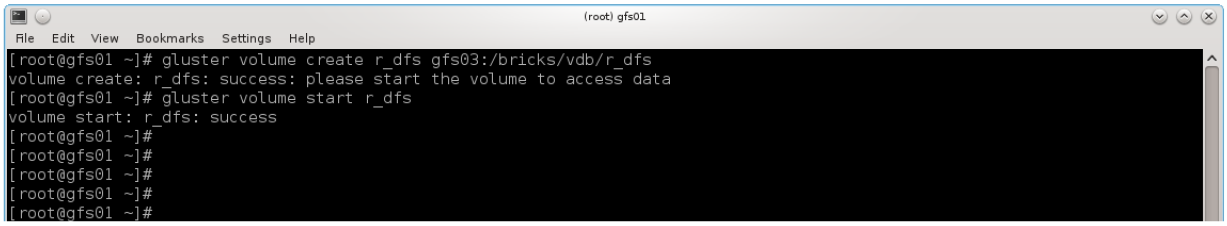
Ara podem veure els tres nodes connectats.

Figura 34: Podem veure l'estat de sincronització entre els tres nodes



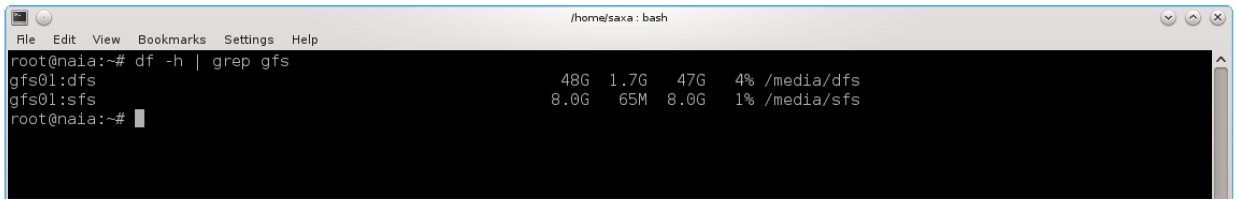
Ara ja podem crear un volum des del node gfs01 però assignant bricks del node gfs03:

Figura 35: Crear r_dfs al node gfs03



I també podem afegir un brick al volum dfs que estigui al node gfs03. Un cop està afegit podem veure com la capacitat del disc ha augmentat:

Figura 36: Augment de capacitat

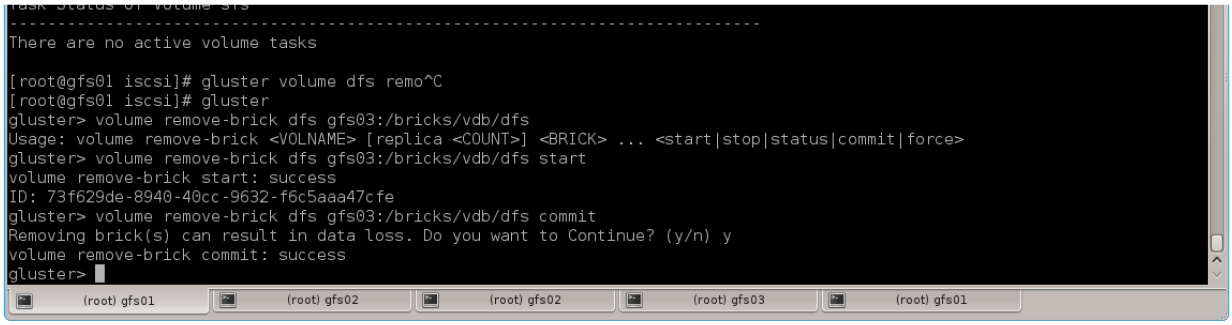


I quin és ara el comportament a l'hora d'escriure dades? Doncs tot i que nosaltres apuntem amb el client FUSE als dos nodes propers, ell intenta balancejar la càrrega entre tots els nodes. Per tant, la velocitat és la del pitjor node, en aquest cas, gfs03.

També he apreciat que moltes operacions poden fallar si no hi ha bona comunicació entre els bricks. Com un sistema distribuït ha d'efectuar operacions en diferents nodes, si una d'aquestes operacions no finalitza bé en algun dels nodes, els que si havien executat l'ordre correctament, poden quedar inestables, havent de corregir la situació baixant a cada un dels nodes. Aquesta inestabilitat de la línia ha provocat haver de repetir força operacions degut a que el node GFS03 perd la comunicació sovint entre algun dels altres nodes.

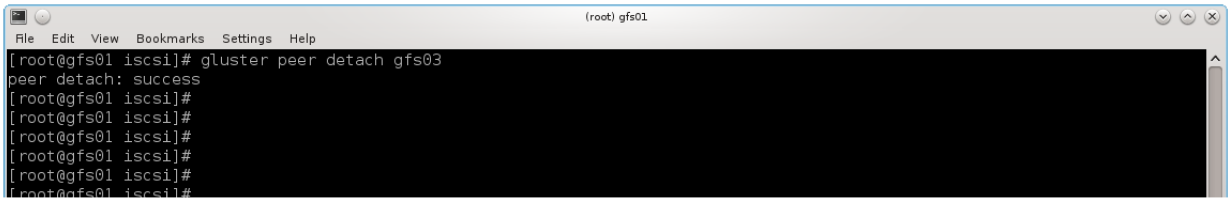
Per tant, ens interessa eliminar el brick. Aquest operació també caldria fer-la quan algun dels nodes implicats quedi desfasat o antiquat.

Figura 37: Eliminar el brick del node gfs03



I finalment, eliminar un node que ja no dóna el rendiment esperat.

Figura 38: Eliminar el node gfs03 del pool



Ara ens caldria definir unes quotes per projectes. Si pensem que tenim un clúster de servidors de moltes Petabytes i no podem controlar on creix la informació tindrem un problema de creixement desorbitat.

Primerament activarem les quotes en els punts de muntatge dels bricks. Un cop fet això podem activar les quotes per a cada volum que vulguem. Les proves que he fet m'han mostrat que, així com a nivell de sistema de fitxers és molt convenient activar les quotes d'usuari o grup abans de començar a posar-hi informació, en aquest cas he vist que l'aplicació de la quota ha estat extremadament bona.

Figura 39: Activació de les quotes sobre el volum DFS

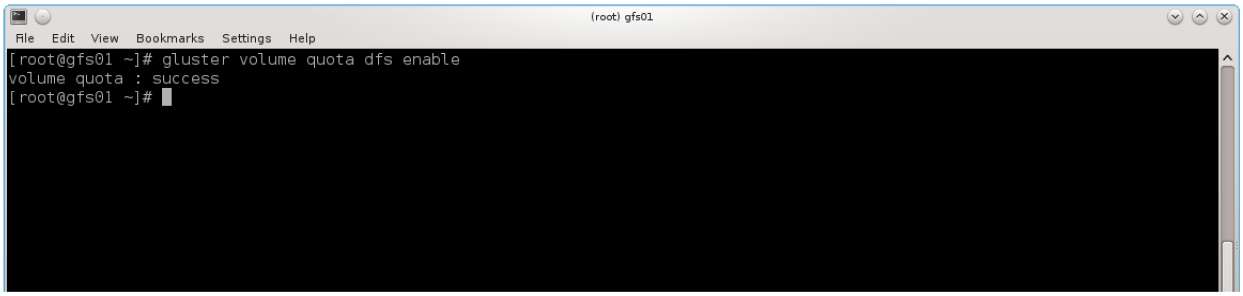
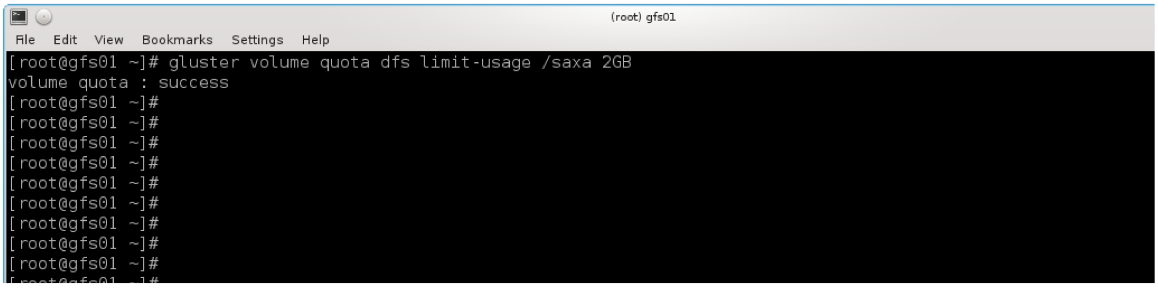
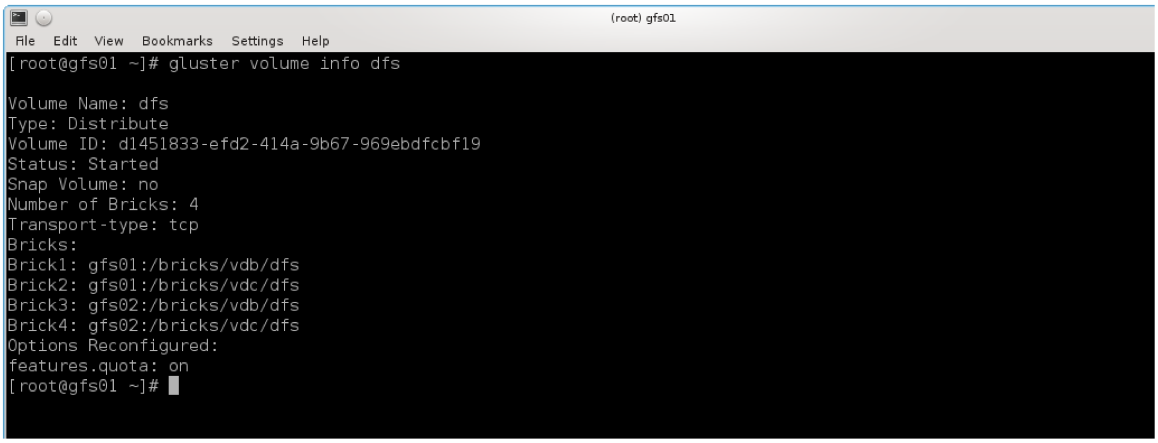


Figura 40: Activació d'una quota sobre un directori



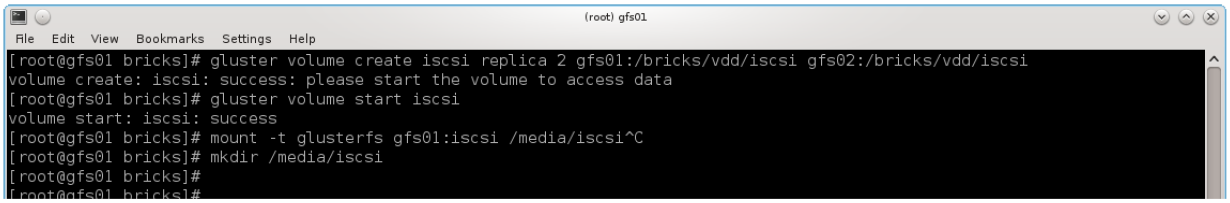
Ara podem veure la configuració del volum i l'estat de les quotes.

Figura 41: Mostrar la configuració del volum DFS



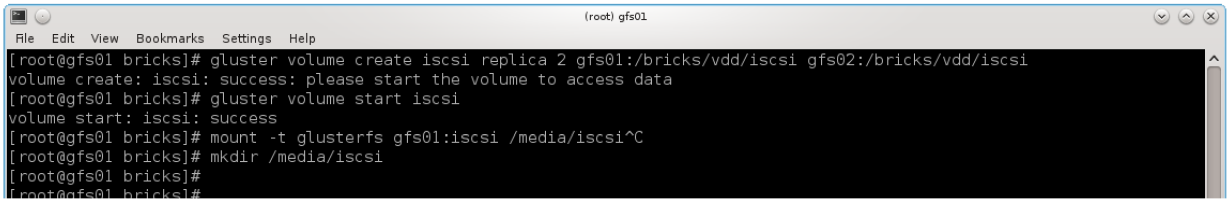
Una altra necessitat que volíem superar era l'accés mitjançant bloc. Per a tal propòsit prepararé un volum replicat on hi ubicaré els fitxers que contindran els discos.

Figura 42: Preparació del volum replicat



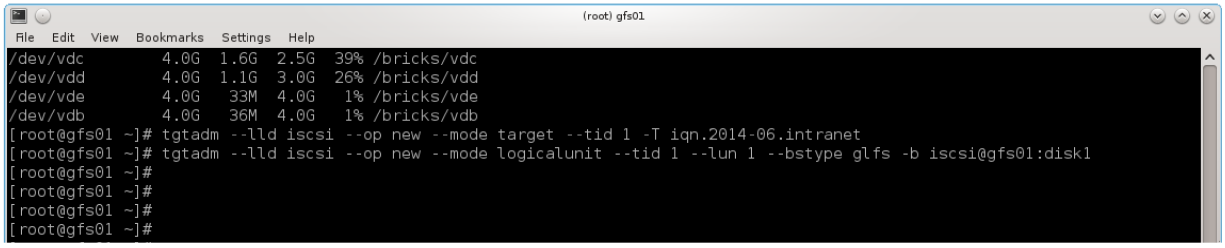
Un cop muntat localment creem un fitxer gran d'una GB amb la comanda "dd". Ara posem en marxa el dimoni del "iscsi-target" modificat per la comunitat de Gluster:

Figura 43: Preparació del volum replicat



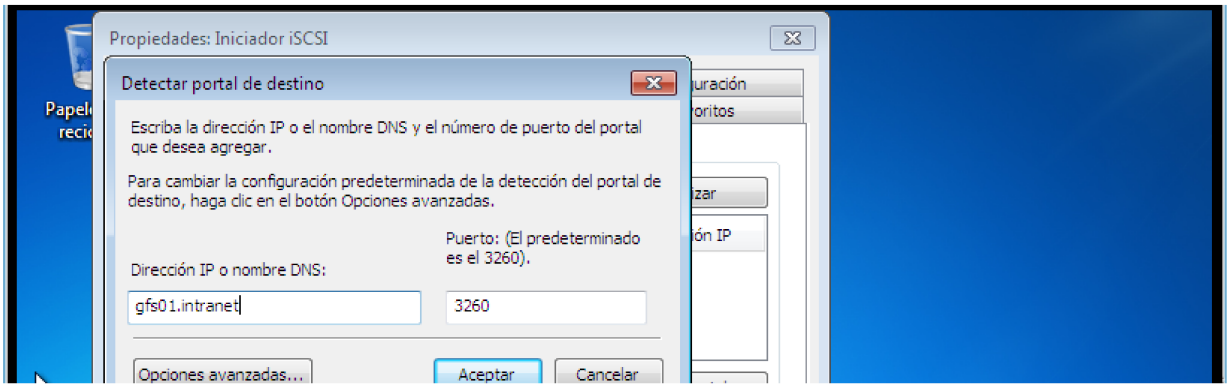
Un cop està disponible el fitxer ja podem executar les comandes per a crear el *target* de l'iscsi.

Figura 44: Execució del dimoni iscsi-target



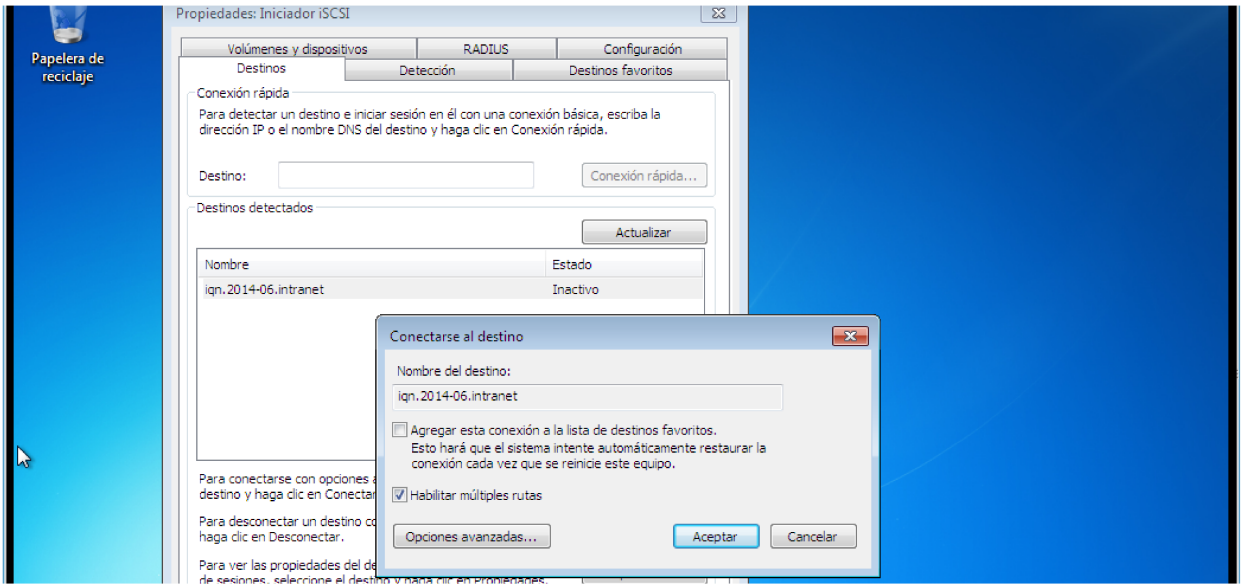
I ara podem veure si tenim disponible el volum amb identificador "iqn.2014-06.intranet" des d'un sistema Windows. Apuntem al servei descobridor i l'afegim.

Figura 45: Connexió d'un sistema Windows per iSCSI



Un cop l'hem afegit a la pantalla de portals ja podem connectar contra els targets.

Figura 46: Connectar al target



I ara ja tenim disponible el disc. Un cop formatat el tenim disponible com una unitat més. Si el volem ampliar... Les proves no han estat bones. Segons alguna documentació es pot ampliar el fitxer, afegint-hi més informació amb la mateixa instrucció "dd". Però les meves proves han estat infructuoses en aquest aspecte. Alternativament es pot crear un nou disc i afegir-lo com una nova LUN. Ha de ser el propi sistema qui permeti l'opció d'unir aquestes LUNs en un únic volum.

He de dir que tot i que les proves han estat molt favorables en aquest entorn, l'estabilitat del dimoni del "tgtd" m'ha decepcionat una mica. Quan he intentat desvincular la LUN per a poder fer operacions sobre el fitxer, el servei de "tgtd" ha provocat un error de segmentació. El problema més greu és que el node ha quedat mig inservible. Ha calgut reiniciar-lo per a poder recuperar-lo adequadament. Evidentment això no podria passar en un entorn productiu real.

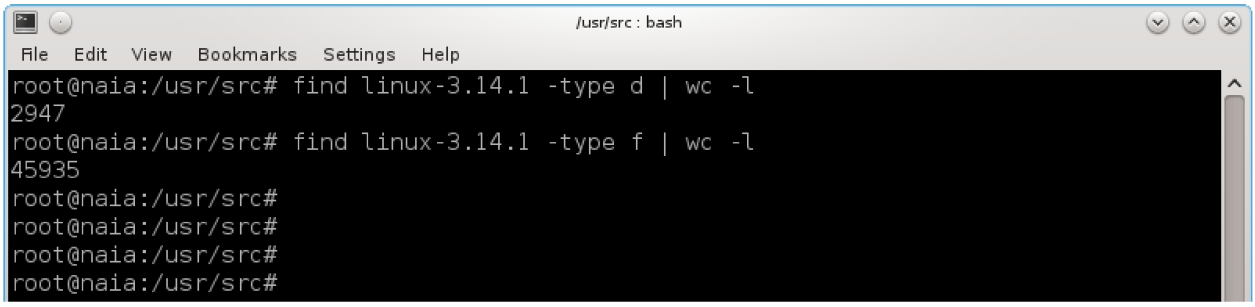
Estadístiques

Executaré un primer set de probes força bàsiques per a veure el correcte funcionament de la infraestructura del pilot:

Trio un fitxer relativament gran però que implica la creació de molts fitxers i directoris. La seva eliminació, com a estructura complexa, també ens mostrarà dades força interessants.
Dades del fitxer: linux-3.14.1.tar.bz2
Tamany: 78400108 bytes
Nombre de fitxers: 45935
Nombre de directoris: 2947

Aquesta descompressió genera una estructura de fitxers i directoris:

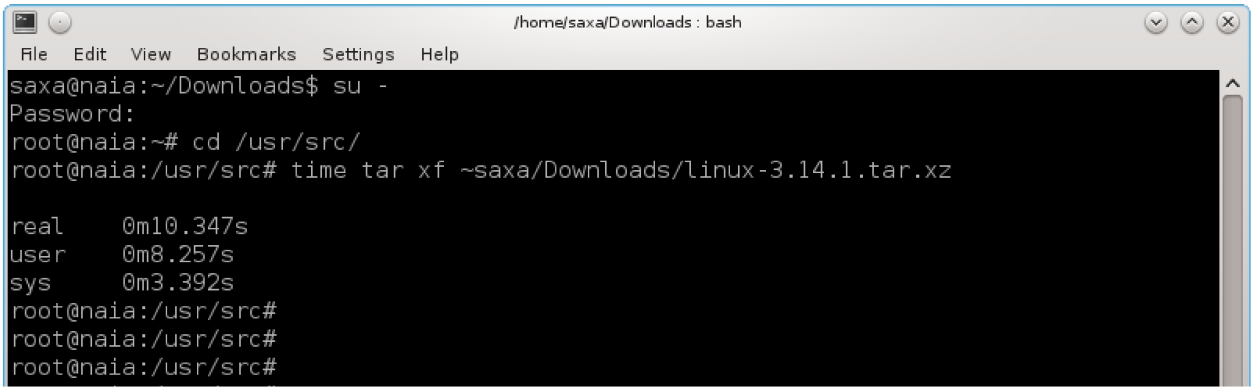
Figura 47: Informació del contingut de les fonts del kernel



Descomprimir tota l'estructura de les fonts del kernel sobre un directori d'un sistema de fitxers locals:

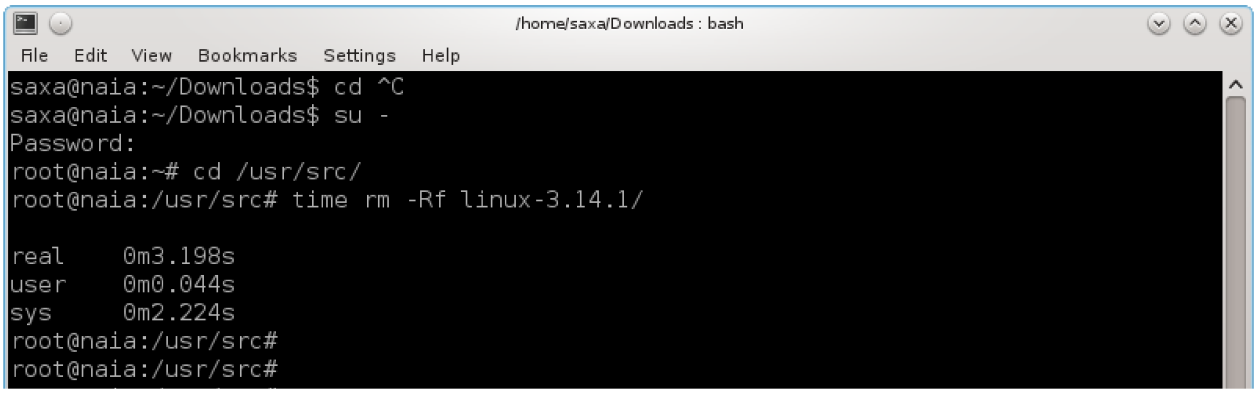
Sobre el host físic:

Figura 48: Descompressió sobre disc físic



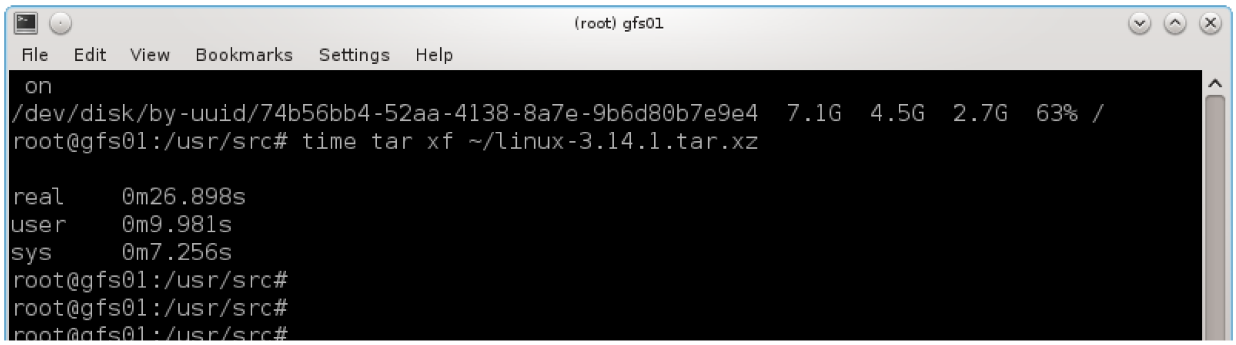
Eliminar tota la estructura prèviament creada sobre el directori local:

Figura 49: Eliminació sobre disc físic



Repetim el mateix procés sobre un disc virtual:

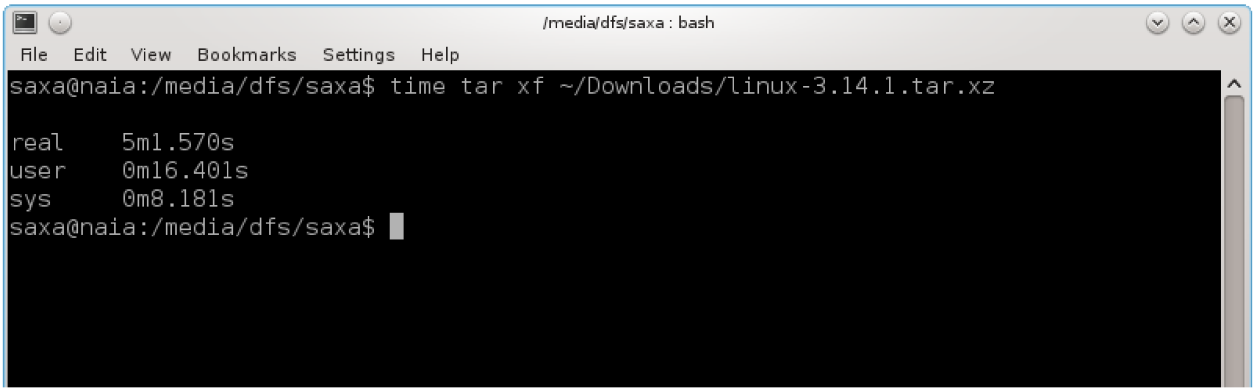
Figura 50: Descompressió sobre disc virtual



En aquest punt estem veien que la descompressió sobre un disc físic (10.3s) i sobre el disc virtual físic (26.9s) ja ens penalitza molt (augment del 261.1%), i caldrà tenir en compte aquesta dada com a referència.

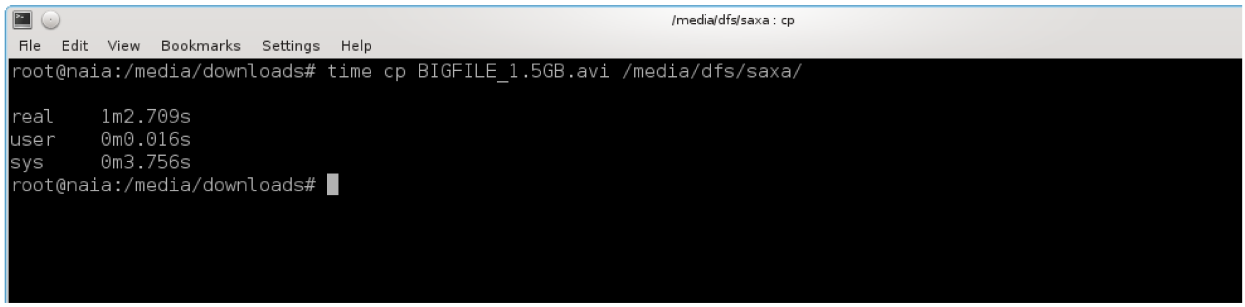
Descomprimir tota la estructura de les fonts del kernel sobre un punt de muntatge de GlusterFS, fent servir FUSE i sobre un volum distribuït:

Figura 51: Descompressió sobre volum virtual distribuït



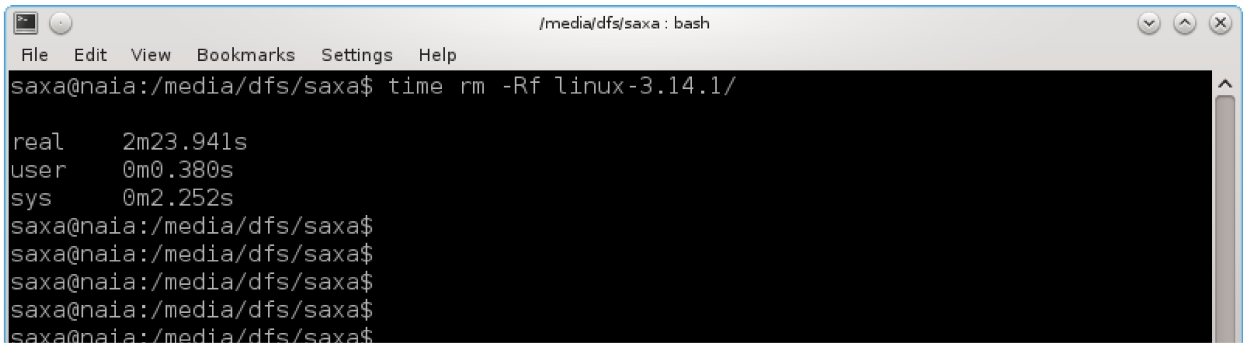
Copiar un únic fitxer de grandària 1.5GB:

Figura 52: Copiar un fitxer gran a volum distribuït



Eliminar tota la estructura prèviament creada sobre el punt de muntatge de GlusterFS:

Figura 53: Eliminació de les dades sobre volum virtual distribuït



I el mateix procés sobre el volum replicat:

Figura 54: Descompressió sobre volum virtual replicat

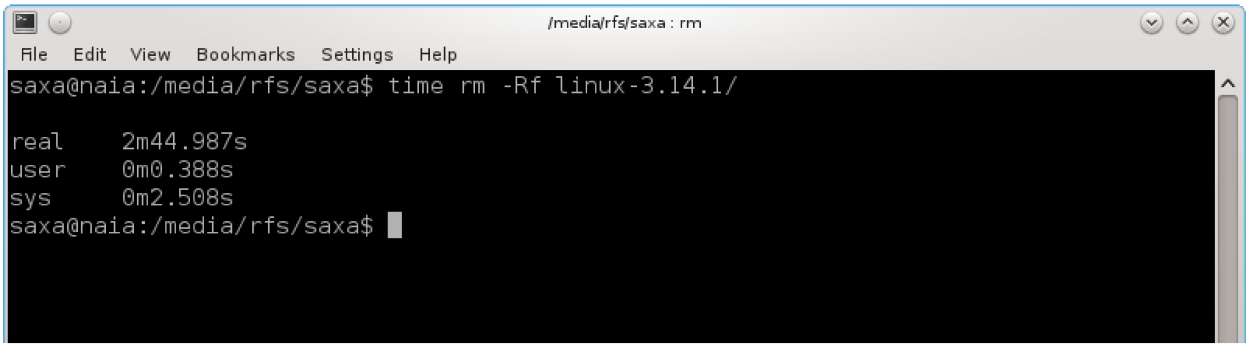
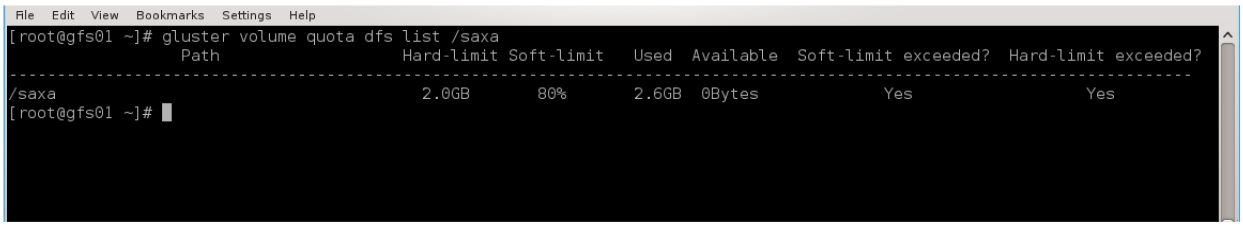


Figura 55: Mostrar l'estat dels límits establers per al volum DFS



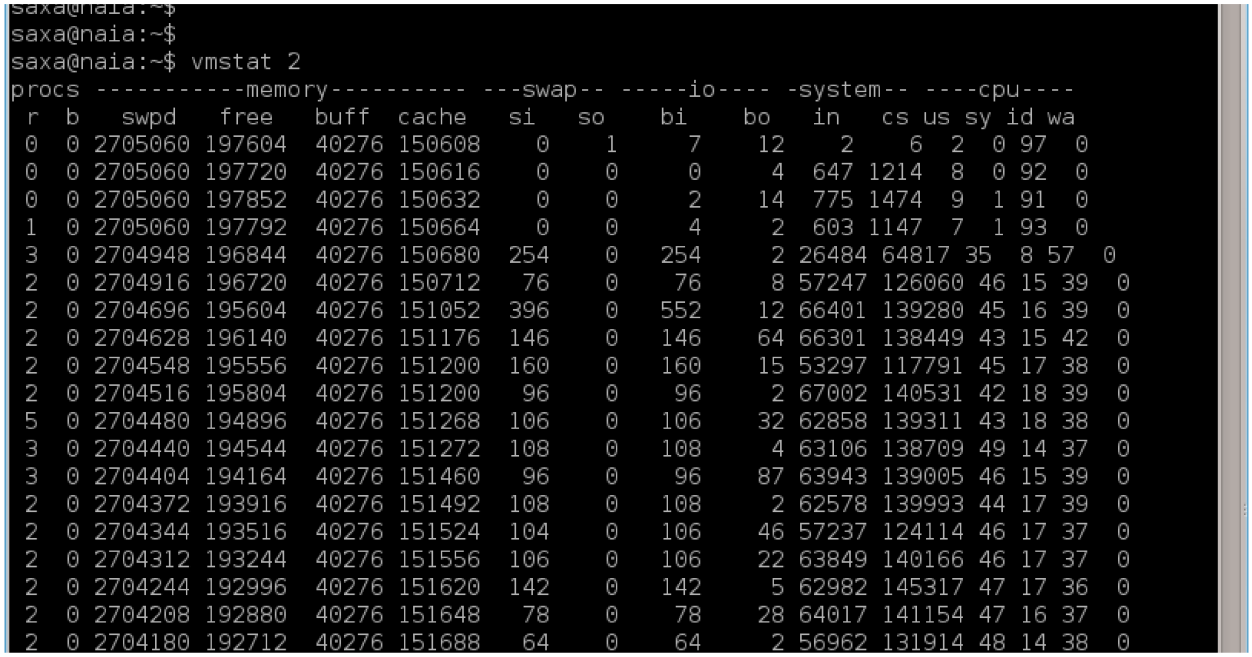
Amb aquestes probes podem veure que el sistema funciona. El rendiment, tot i que això no és el que s'està avaluant actualment, veiem que ens pot sorprendre pel pobre resultat que dóna, però l'entorn no és l'òptim per a fer un *benchmark* i el tipus de fitxers que estem tractant no és el més adequat per a un entorn distribuït.

Mirem d'aprofundir una mica més utilitzant les eines que disposem a Internet per a efectuar probes d'estrès i rendiment.
Un cop vist aquest set de probes podem veure els resultats d'eines d'estrès més eficients com iozone.

IOZone s'ha compilat sobre el host principal i executaré els tests sobre els diferents punts de muntatge virtuals.

Abans de veure els resultats he de fer una apreciació. Quan vaig començar a efectuar els tests d'estrès, vaig poder apreciar que els percentatge d'ús dels processadors augmentava considerablement. A la següent figura podem veure la sortida de la comanda "vmstat" durant l'execució d'un dels tests:

Figura 56: Ocupació CPU



El que vull denotar és com l'ús dels processadors augmenta considerablement a partir de la cinquena iteració. Veiem com el percentatge d'ús augmenta fins,

aproximadament, el 50%, que curiosament són els dos processadors assignats a les màquines virtuals (de les quatre disponibles). Aquest augment d'ús respon exactament a l'inici d'un dels tests de IOZone. Com els processos que consumeixen vist des de la banda del host són el processos “kvm”, és indicador que realment està consumint el processador el servidor de Gluster. Per tant, si algun dia ens plantegem posar en marxa una clúster basat en Gluster haurem de tenir en compte que els processadors dels bricks són força importants. No hauríem de pensar només en posar discos ràpids si els processadors no seran capaços d'aguantar la càrrega.

Un cop executat uns quants test bàsics procedeixo a executar unes proves més intensives amb un programa d'estrès de discs. L'aplicació s'ha executat sobre el host principal sobre els punts de muntatge DFS (el recomanat per RedHat) i el, teòricament més preparat per a entorns intensius, SFS (stripped).

Estadístiques d'ús sobre el volum distribuït i stripped. Només per a tenir algun valor de referència, ja que, com he dit abans, un entorn virtualitzat no és el millor entorn per a fer un benchmark.

Write Speed

Figura 57: IOZONE – Write Speed sobre disc distribuït

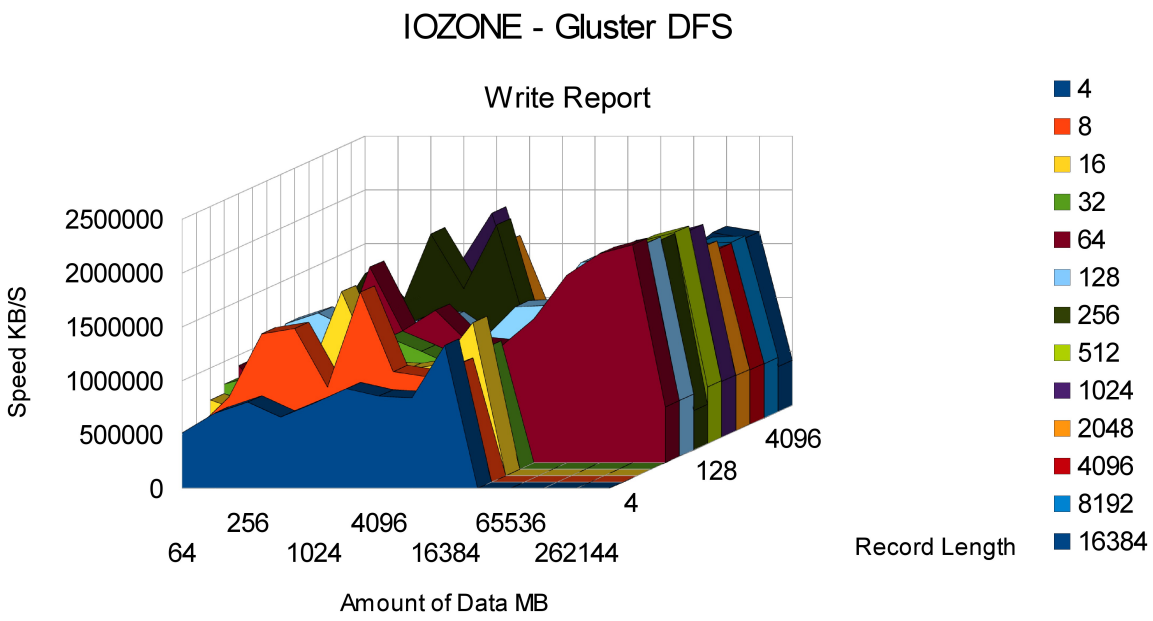
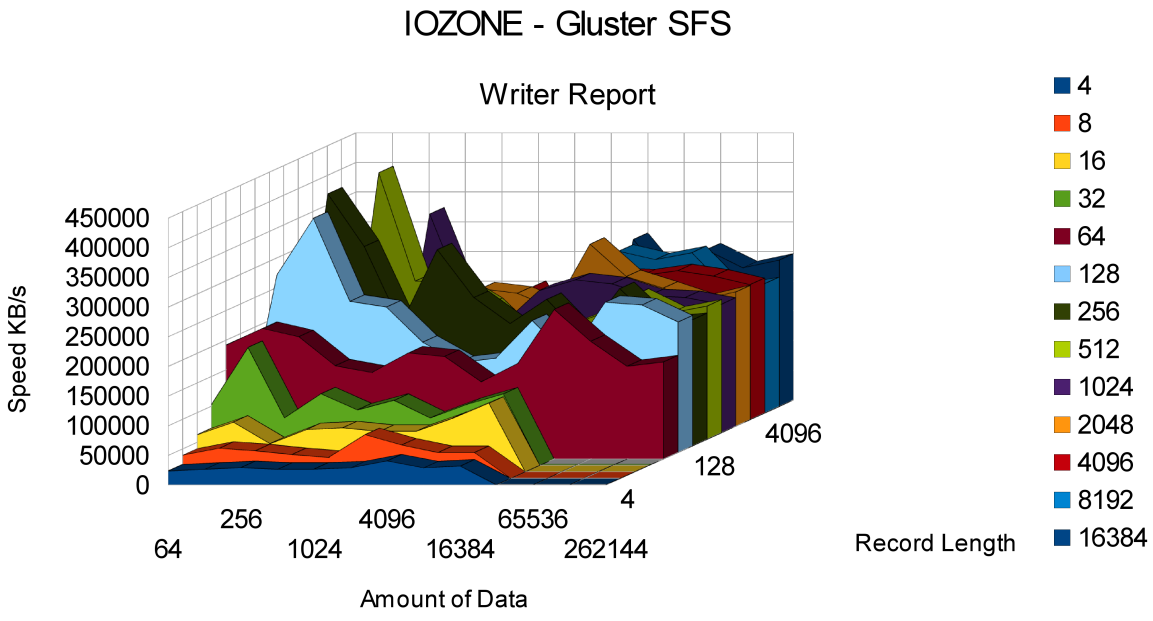


Figura 58: IOZONE – Write Speed sobre disc Stripped



Re-write Speed

Figura 59: IOZONE – RE-Write Speed sobre disc distribuït

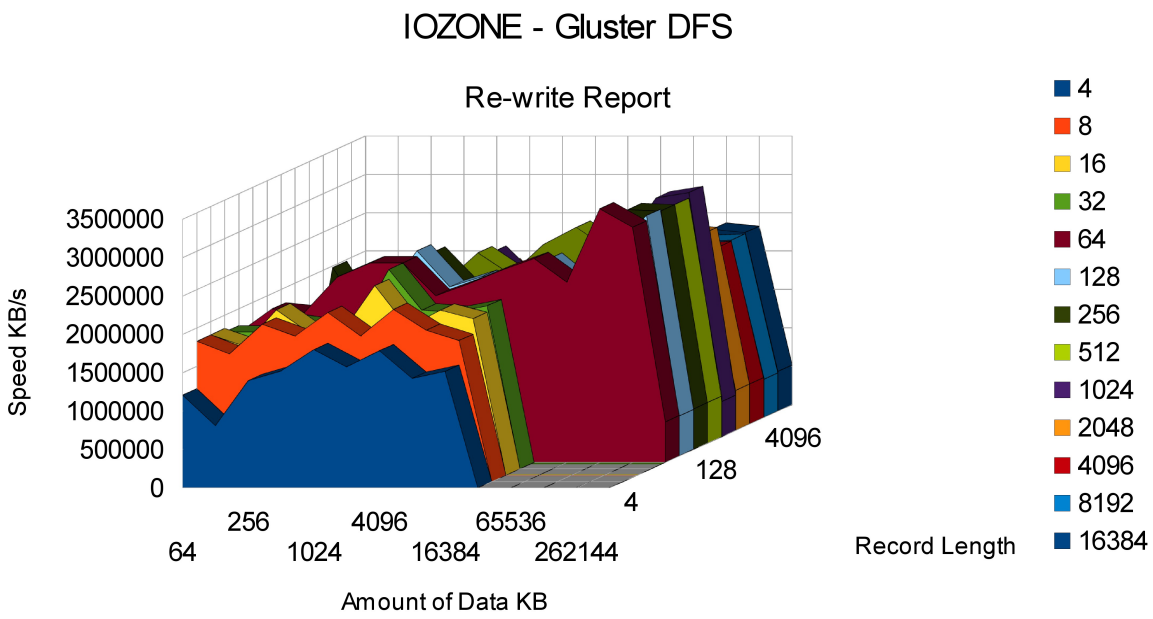
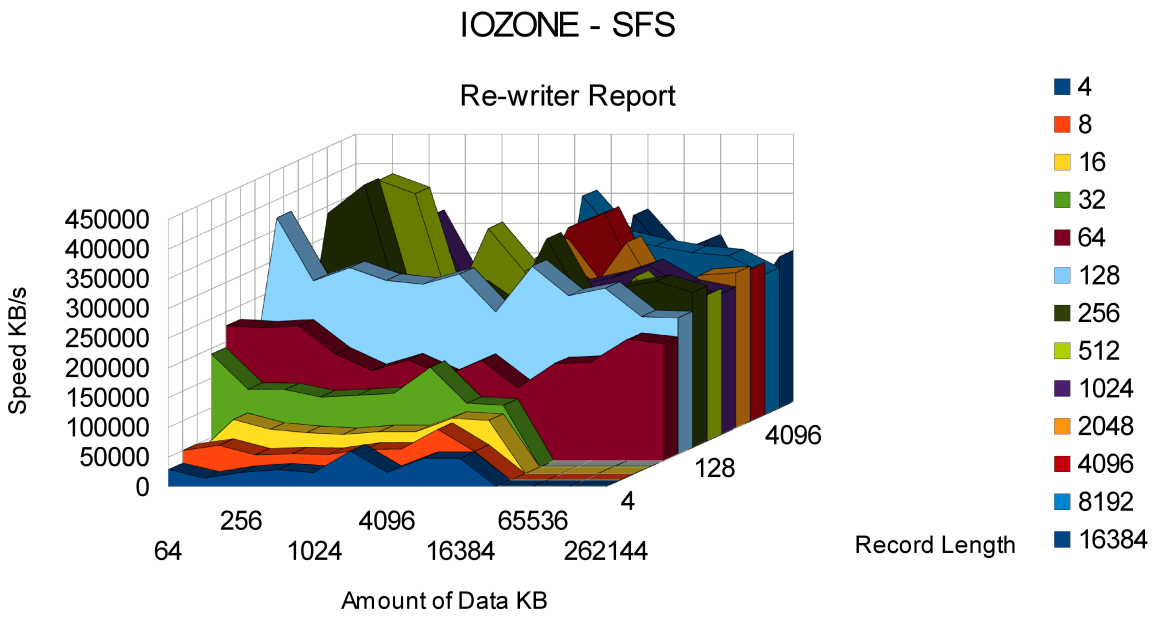


Figura 60: IOZONE – RE-Write Speed sobre disc Stripped



Read Speed

Figura 61: IOZONE – Read Speed sobre disc distribuït

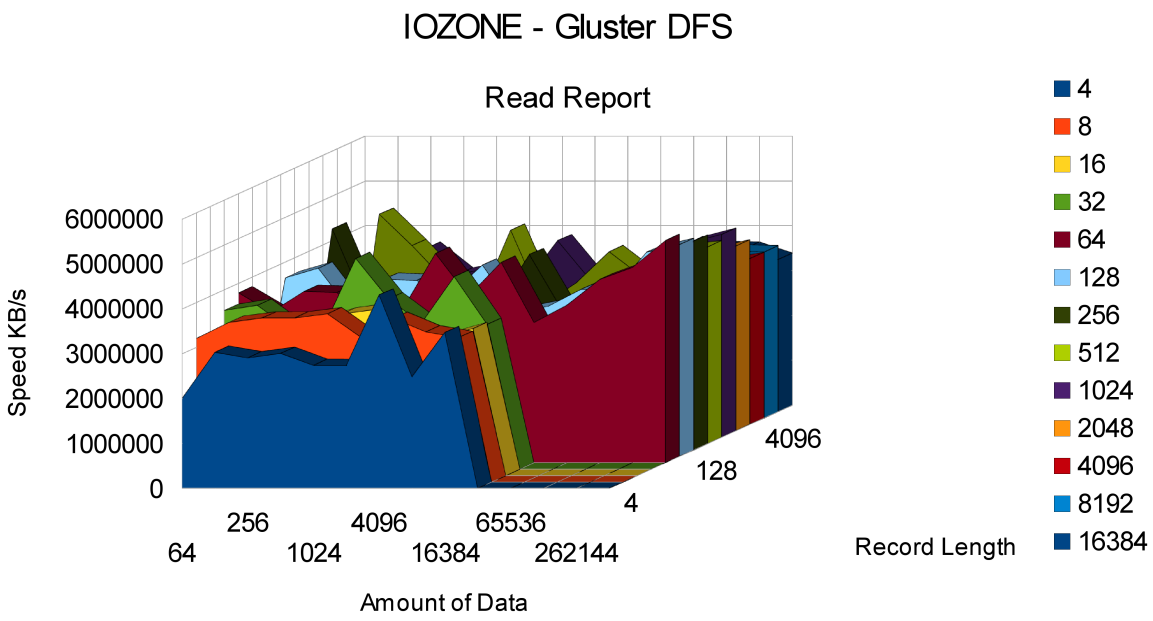
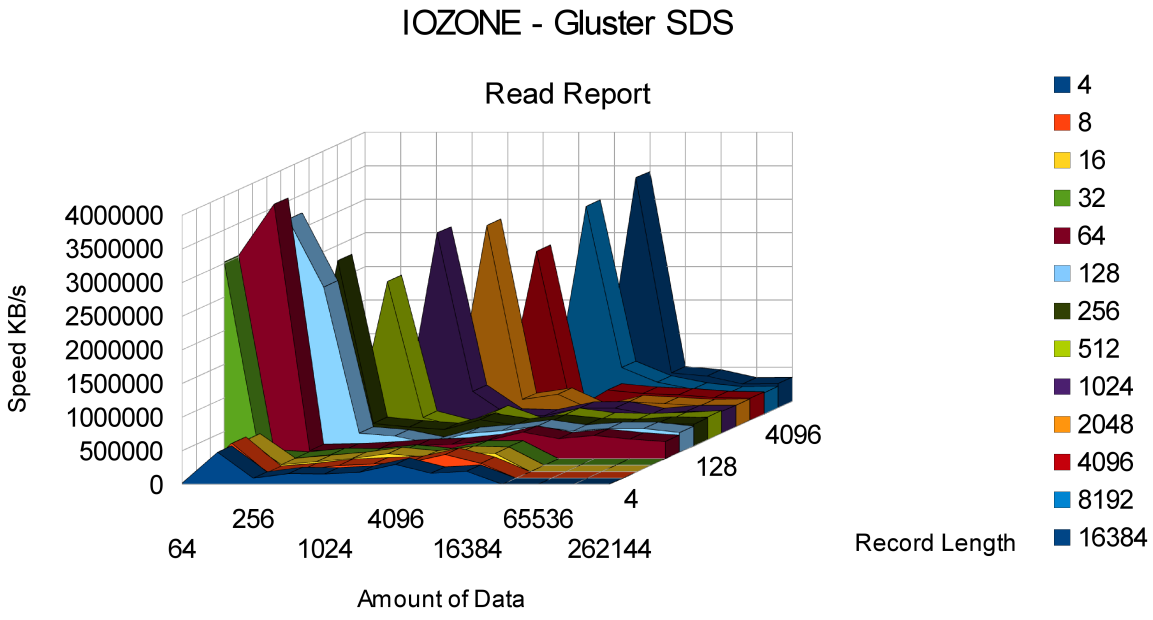


Figura 62: IOZONE – Read Speed sobre disc Stripped



Re-read Speed

Figura 63: IOZONE – Re-Read Speed sobre disc distribuït

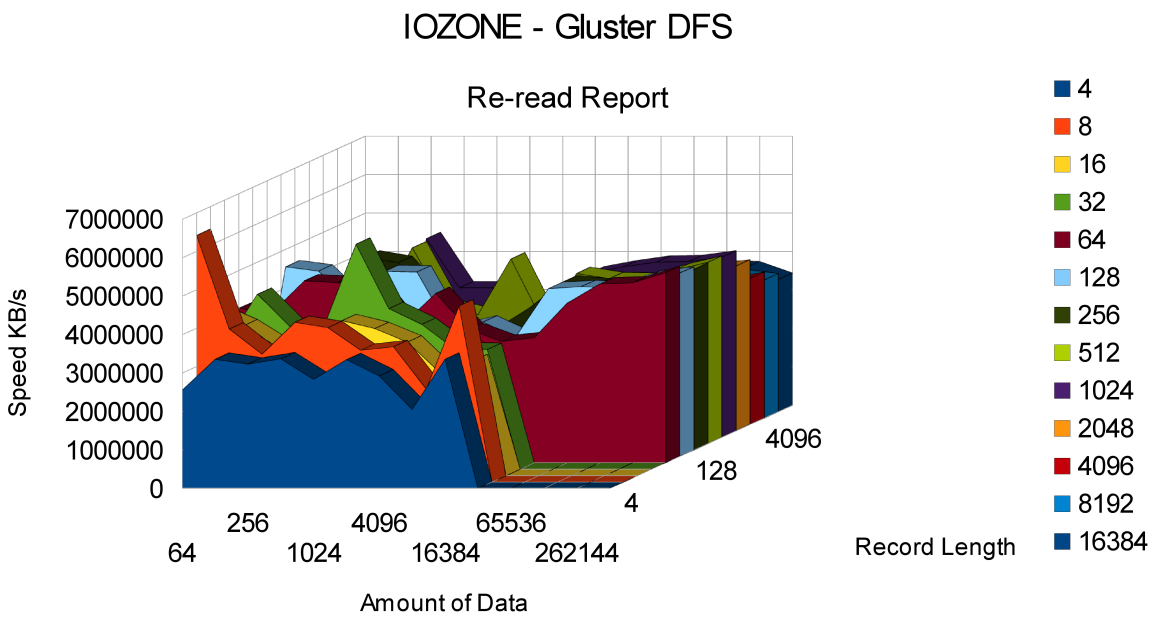
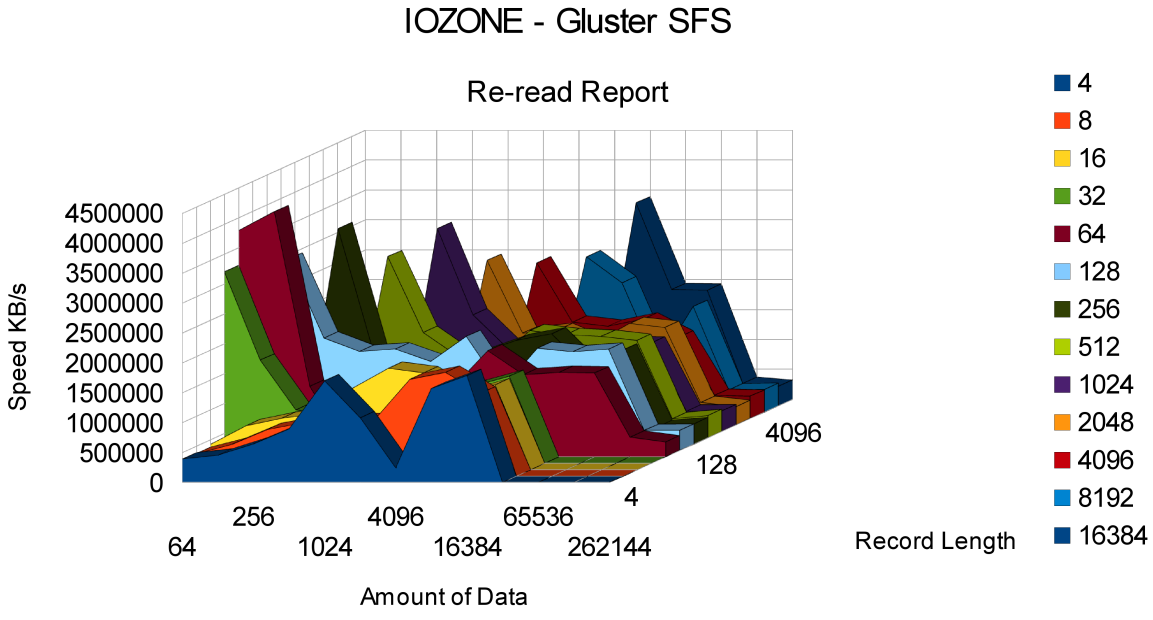


Figura 64: IOZONE – Re-Read Speed sobre disc Stripped



fread Speed

Figura 65: IOZONE – fRead Speed sobre disc distribuït

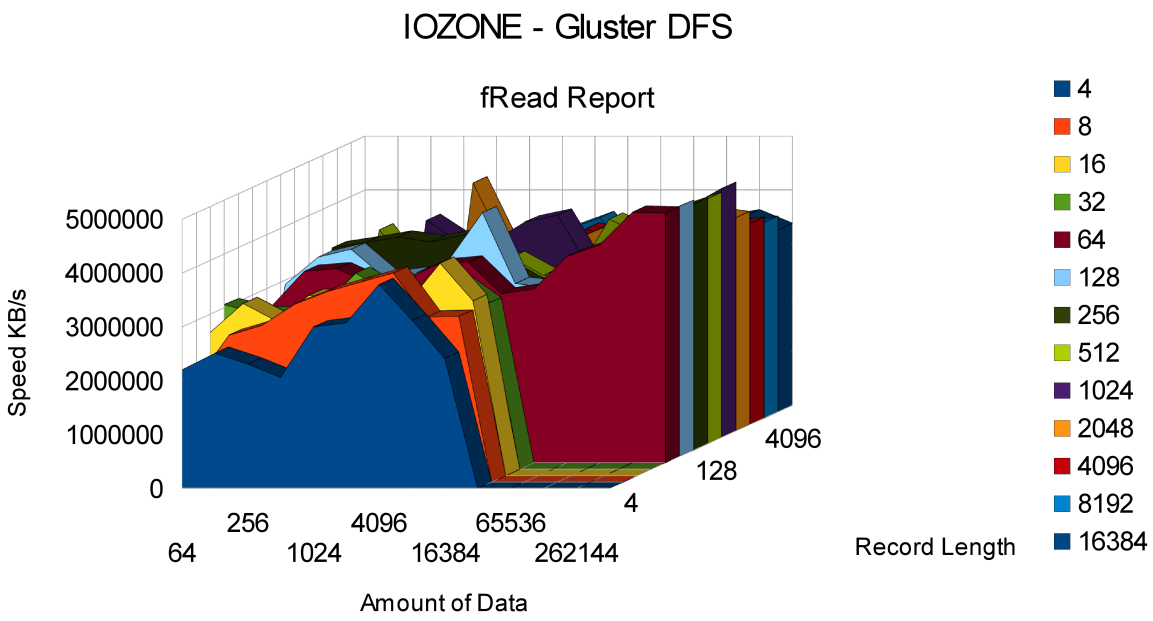
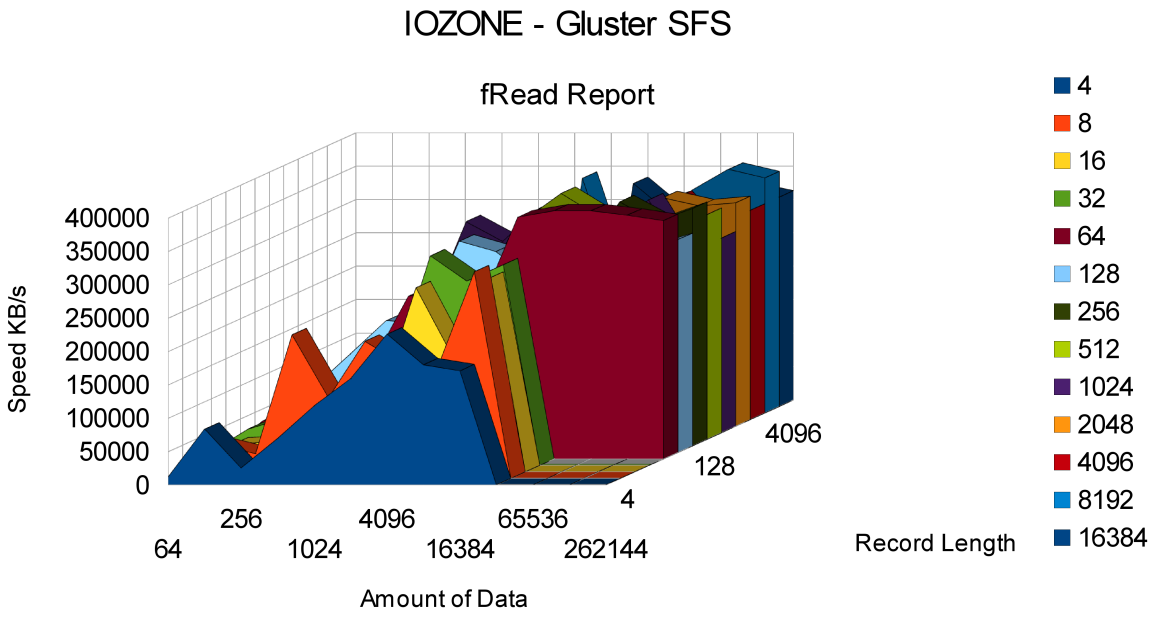


Figura 66: IOZONE – fRead Speed sobre disc Stripped



RE-fread Speed

Figura 67: IOZONE – RE-fRead Speed sobre disc distribuït

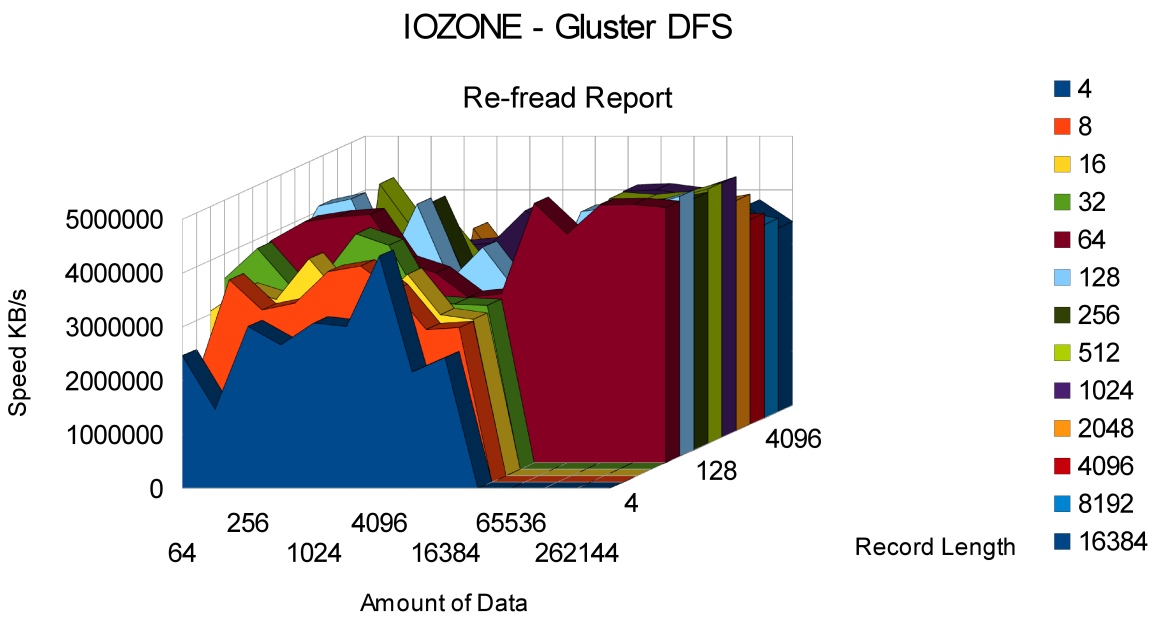
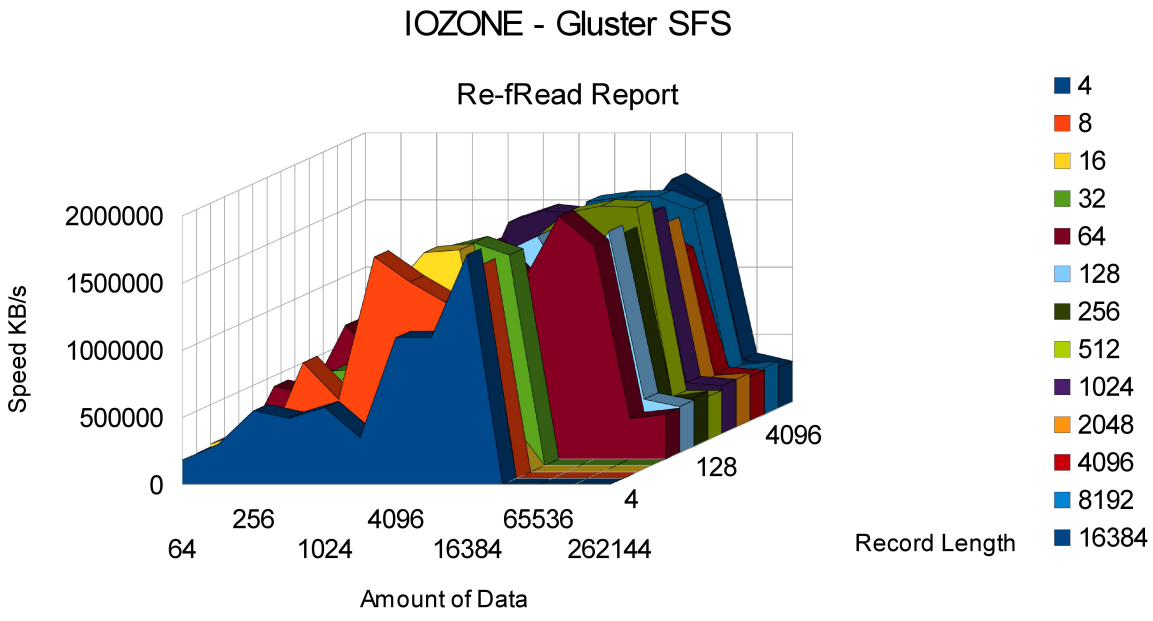


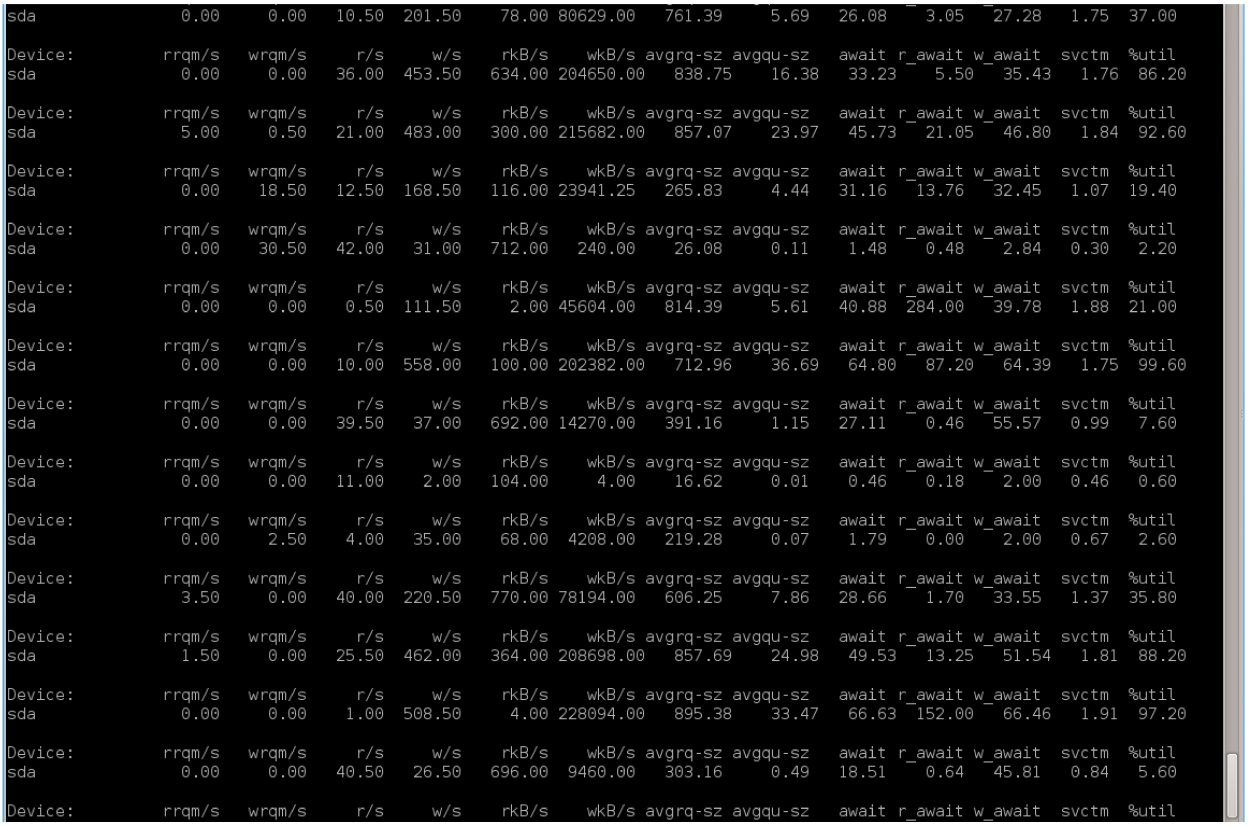
Figura 68: IOZONE – RE-fRead Speed sobre disc Stripped



També podem veure les estadístiques del disc físic mentre estava executant aquests test, per a una mostra:

Podem apreciar com “iostat” ens mostra índex d'escriptures de fins a 228Mb/s, que s'acosta als límits dels 3Gbps d'ample de banda del canal SATA-2.

Figura 69: IOSTAT sobre l'equip físic



Conclusions

Abans de treure conclusions hauríem de veure que fan els fabricants propietaris, ja que, ara per ara són els predominants en les solucions empresarials, i segurament ens poden mostrar si les solucions Open Source van en la bona direcció.

Què fan els fabricants de solucions hardware?

HITACHI[13]

L'última família de productes d'Hitachi són els sistemes HUS (Hitachi Unified System) i la seva arquitectura es basa en tenir diferents productes per a cada necessitat. Tenen el seu sistema clàssic d'accés al sistema de bloc mitjançant connexions FC, FCoE o iSCSI, un altre producte anomenat HNAS per a oferir la solució de NAS, i encara un altre producte HCP (Hitachi Content Platform) per a l'accés a objectes.
La seva solució es basa en tenir un gran repositori de dades i els diferents productes per a cada servei, fent servir el millor de cada producte per a cada àmbit.

EMC[14]

EMC també diversifica el seu producte mitjançant la cabina clàssica per a l'accés en mode bloc, i un producte anomenat Isilon per al sistema de NAS. Aquest producte és la resposta Scale Out per a un sistema distribuït d'alt rendiment i gran capacitat. Aquest solució tan diferenciada entre un producte i l'altre aporta tenir el millor de cada món per a cada necessitat. Però també aporta el pitjor de cada món, i el tenir un *expertise* específic per a cada solució, i una distribució diferent del disc, no poden integrar ambdues solucions per, o bé re-aprofitar els recursos ociosos, o bé per a unificar la gestió i coneixement.

EMC també disposa d'un producte (ViPR Software Defined Storage) que fa gala de la nova tendència del DCaaS (DataCenter as a Service). El concepte d'aquesta solució és crear una capa d'abstracció del maquinari, permeten vincular-se amb diferents fabricants (com ara NetAPP) i solucions de emmagatzemament de cloud, com ara OpenStack (Swift). Sembla una solució molt ambiciosa i interessant, però no oblidem que això ens aporta una capa de virtualització al damunt del nostre storage. Així no tenim més remei que seguir buscant les nostres solucions específiques per a l'emmagatzemament.

NetAPP[15]

Network Appliance és l'únic que té una plataforma unificada per a l'accés a les dades. En el seu mateix producte ofereixen múltiples canals d'accés i protocols. El seu sistema Clustered Data ONTAP ofereix la possibilitat de gestionar l'emmagatzemament en una única plataforma distribuïda en un nombre limitat de nodes. La seva solució

13 Hitachi Data Systems (http://www.hds.com)
14 EMC (http://spain.emc.com/storage/index.htm?nav=1)
15 NetAPP (http://www.netapp.com/es/)

permet aprofitar els recursos gestionant-los i reassignant-los allà on sigui necessari. Les seves limitacions de grandària màxima per volum no són especialment grans (fins a 324TB en un únic volum a la seva família *High Level*)

El que n'extrec d'aquesta informació és que difícilment hi ha una solució única per a tots els entorns. Com hem vist a l'apartat d'anàlisis, les necessitats d'un entorn són molt diferents les unes de les altres.

Ara bé, després de fer totes les proves sobre un petit pilot de Gluster he de dir que, tot i que he quedat gratament sorprès, no el veig com una solució que actualment es pugui posar en producció com una plataforma global.

Valoració de les diferentes opcions d'accés mitjançant els diferents protocols.

Tot i no poder provar protocols sobre F*iber Channel* (FCoE) les funcionalitats d'accés mitjançant iSCSI han estat força bones. El rendiment ha estat prou bo. El que manca és una interfície de gestió on qualsevol persona sense un coneixement tant profund de la solució pogués administrar-ho amb un relativa simplicitat.
També cal afinar els serveis de iscsi-target ja que han demostrat no ser tot lo estables que caldria. La comunitat de Gluster, coneixedora que pot ser una solució per a tal propòsit, sembla que hi estigui dedicant esforços a estabilitzar la solució. Ara mateix, com ja hem vist, traspassen una capa de sistema de fitxers per a parlar amb el volum de Gluster directament sense passar per la capa de sistema de fitxers. A més de provocar un augment de rendiment permet la generació de snapshots i la sincronització entre clústers remots.

Del client FUSE natiu poca cosa cal dir. El seu rendiment i funcionalitat és molt bo. Evidentment el client natiu proporciona un accés paral·lel a tots els nodes que allotgen el volum. Amb això el rendiment és lineal, a més nodes, més rapidesa. En aquest punt cal remarcar que les configuracions possibles són força bones, i a la propera versió 3.6 n'hi afegiran de noves. Tot i que el rendiment no era l'esperat cal tenir en compte que l'entorn no és l'apropiat per a un *benchmark.*

Per la part de NFS cal dir que encara no està prou madur. La implementació de la versió NFSv4 encara està al *roadmap*. I un punt molt favorable que aporta NFSv4 és el Parallel NFS, on el client pot accedir concurrentment a diversos nodes i la integració amb sistemes Kerberos. L'actual sobre NFSv3 ja implementa les ACLs i és molt estable. Proporciona les configuracions necessàries per a pràcticament qualsevol entorn.

La solució per als clients de Windows, tot i no ser tant potent com la del client natiu, és força bona on podem arribar a definir el SAMBA amb un volum replicat. Aquesta configuració permet, en entorns d'alta concurrència d'usuaris, balancejar la càrrega entre els nodes. Però, evidentment, no podrem aconseguir més rendiment del que ens pot oferir un únic node.

L'entorn és força senzill de configurar. Les opcions que permet estan força controlades i abans de cometre qualsevol acció perillosa t'avisa.
Snaphsots. Aquest punt m'ha decepcionat molt. Tot i que des de la versió 3.4 Gluster implementa les snapshots de volum, la novetat de la versió 3.5 són les snapshots de fitxers. Aquesta funcionalitat ve requerida per la realització de snaps d'imatges. Aquestes imatges poden ser de màquines virtuals o volums servits per iSCSI.

Ho he intentat amb paquets oficials de Gluster, amb les fonts compilades de diferents versions, i no ha funcionat. Sembla que hi ha un bug (1094815) i no funciona correctament, o potser l'entorn és massa reduït per a poder aplicar aquest tipus de configuracions més avançades.

Aquest punt és bàsic per a una possible implantació d'un clúster gran. Realitzar còpies diàries d'un volum d'informació tant gran serà força difícil. També cal tenir en compte que les màquines virtuals s'han de posar en un estat coherent per a poder fer-ne una còpia, i sense aquest funcionalitat, caldrà aturar la màquina si volem coherència de dades.

Les funcionalitats bàsiques que buscàvem no han estat del tot aconseguides. Com bé apuntava les snapshots no estan disponibles. I RedHat recomana que sempre es faci servir volums distribuïts i gestionar la replicació entre ells. Aquest punt cal tenir-lo en compte per a la possible dispersitat de les dades. I si els nostres clients són molt intensius un a un tindrem un problema amb el rendiment.

Referent a la ubicació dels volums quan ho mirem amb una mica de deteniment el primer que podem veure és que definir la ubicació de cada segment o rèplica de cada volum serà un mal de cap (split-brain) per a gestionar, i on ha de residir cada còpia per garantir-ne la integritat en cas de fallida. Per això la gent de Gluster han implementat una funcionalitat nova des de la versió 3.4. Aquesta sembla força interessant: Server-quorum. La funcionalitat especifica la quantitat de vegades que ha d'estar replicada una informació, de tal manera que no cal pensar on i quins bricks cal replicar. Amb aquest paràmetre garantim dinàmicament la integritat.

Les accions bàsiques d'afegir bricks als volums funcionen molt bé, sempre i quan tinguem bona comunicació entre els nodes. Un cop hem afegit un brick, sabem que disposem de més capacitat, però des d'aquell moment, la informació no està ben balancejada. Així que disposem d'una eina per a re-balancejar-la. Hem de ser conscients que qualsevol procés de re-balanceig serà costosa per al sistema, sobre tot si és un gran clúster.

He trobat a faltar un *tiering* per volum. Seria molt interessant disposar d'aquesta funcionalitat per alleugerir costos en el maquinari.

Les eines de monitorització són massa tècniques i poc efectives. Descobrir què està fent el nostre clúster és força complexe. No he entrat a provar-les, però són força inintel·ligibles. Seria força interessant que la comunitat preparés un projecte on es pogués veure d'una forma més clara on està la càrrega, on s'està portant a terme la principal feina, la ocupació dels bricks, el percentatge d'ús dels processadors, els problemes de baixar les dades al disc, quina part de consum provoca la redundància, etc.

També cal tenir en compte una qüestió que, estant fora de l'àmbit tècnic, és molt important. Al sistema de storage hi emmagatzemem tota la informació vital per a la nostra empresa. Avui en dia la majoria d'empreses disposen de tècnics de primer nivell que s'encarreguen de les tasques diàries d'administració de la cabina. Quan cal fer alguna operació que no és habitual s'acostuma a contactar amb el segon nivell que és un suport extern, normalment el fabricant o *partner* professional que té unes certificacions del producte. A més, hi ha un canvi de paradigma a l'hora de gestionar i administrar aquests sistemes. I cal guanyar una confiança que als fabricants de

maquinari específic de storage se'ls hi suposa, més que probablement per que l'han aconseguit al llarg dels anys.

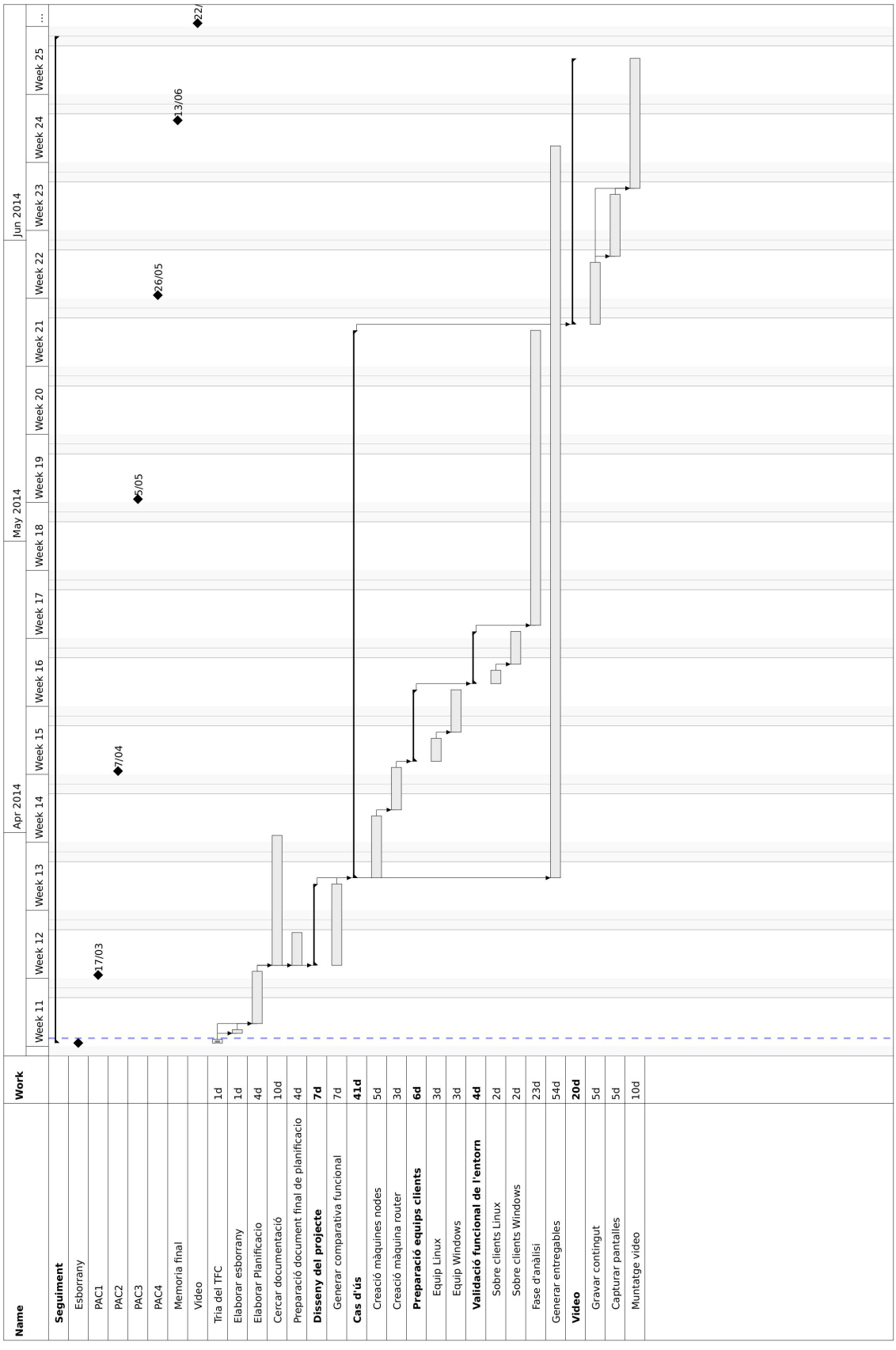
També cal tenir en compte que aquestes solucions, tot i ser Open Source, no són gratuïtes. Tenen cost, potser no en llicències, però si en implantació. A més hem de tenir en compte que estem comprant una solució per tenir discos. Els SDS ens proporcionen la intel·ligència, però l'equipament físic cal comprar-lo, ja sigui una cabina de discos o uns nodes dedicats a fer de bricks.

Així puc concloure que, tot i que sigui una opció molt prometedora i interessant, encara no la veig com una solució global que cobreixi les necessitats de:

- Donar cabuda a tots els protocols d'accés necessaris per a una empresa.
- Permetre una replicació mitjançant comunicacions de baix cost.
- Aconseguir una solució *backup free* per a grans volums de dades.

Annexos

Pla de treball



Bibliografia

- Documentació oficial de Gluster:
  - Versió 3.4:

    http://www.gluster.org/wp-content/uploads/2012/05/Gluster_File_System-3.3.0-Administration_Guide-en-US.pdf
  - Versió 3.5:

    https://github.com/gluster/glusterfs/tree/master/doc/admin-guide/en-US/markdown
  - Roadmap versió 3.6:

    http://www.gluster.org/community/documentation/index.php/Planning36
- Documentació oficial de CEPH:
  - http://ceph.com/docs/master/
- Documentació oficial de XtreemFS:
  - http://www.xtreemfs.org/xtfs-guide-1.5/index.html
- Documentació oficial de TahoeLAFS:
  - https://tahoe-lafs.org/trac/tahoe-lafs/wiki/Doc
- Documentació oficial d'OpenAFS:
  - http://docs.openafs.org/AdminGuide/index.html
- Documentació d'OpenStack (cinder, glance i swift)
  - http://www.openstack.org/software/openstack-storage/

Taula de figures

Referències WEB

- www.gluster.org
- www.openstack.org
- www.xtreemfs.org
- https://ceph.com/docs/master/cephfs/
- http://www.slideshare.net/Inktank_Ceph/scaling-ceph-at-cern
- http://www.slideshare.net/Inktank_Ceph/fj-20140227-cephbestpractisedistributedintelligentunifiedcloudstoragev4ksp
- http://www.slideshare.net/ManfredFuruholmen/osdc?qid=3fda30c1-6d87-4fea-a3af-5a60beefc4fa&v=default&b=&from_search=4
- http://oithelp.nd.edu/applications-and-operating-systems/open-afs/
- http://major.io/2010/08/11/one-month-with-glusterfs-in-production/
- http://www.slideshare.net/keithseahus/creating-a-shared-storage-service-withglusterfsnttpc
- http://edoc.hu-berlin.de/dissertationen/stender-jan-2013-01-04/PDF/stender.pdf
-