Implementazione di LUSTRE File System per Cluster di Calcolo OAS

Autore: Alessandro Tacchini, INAF OAS Bologna

Indice

1. Perchè LUSTRE

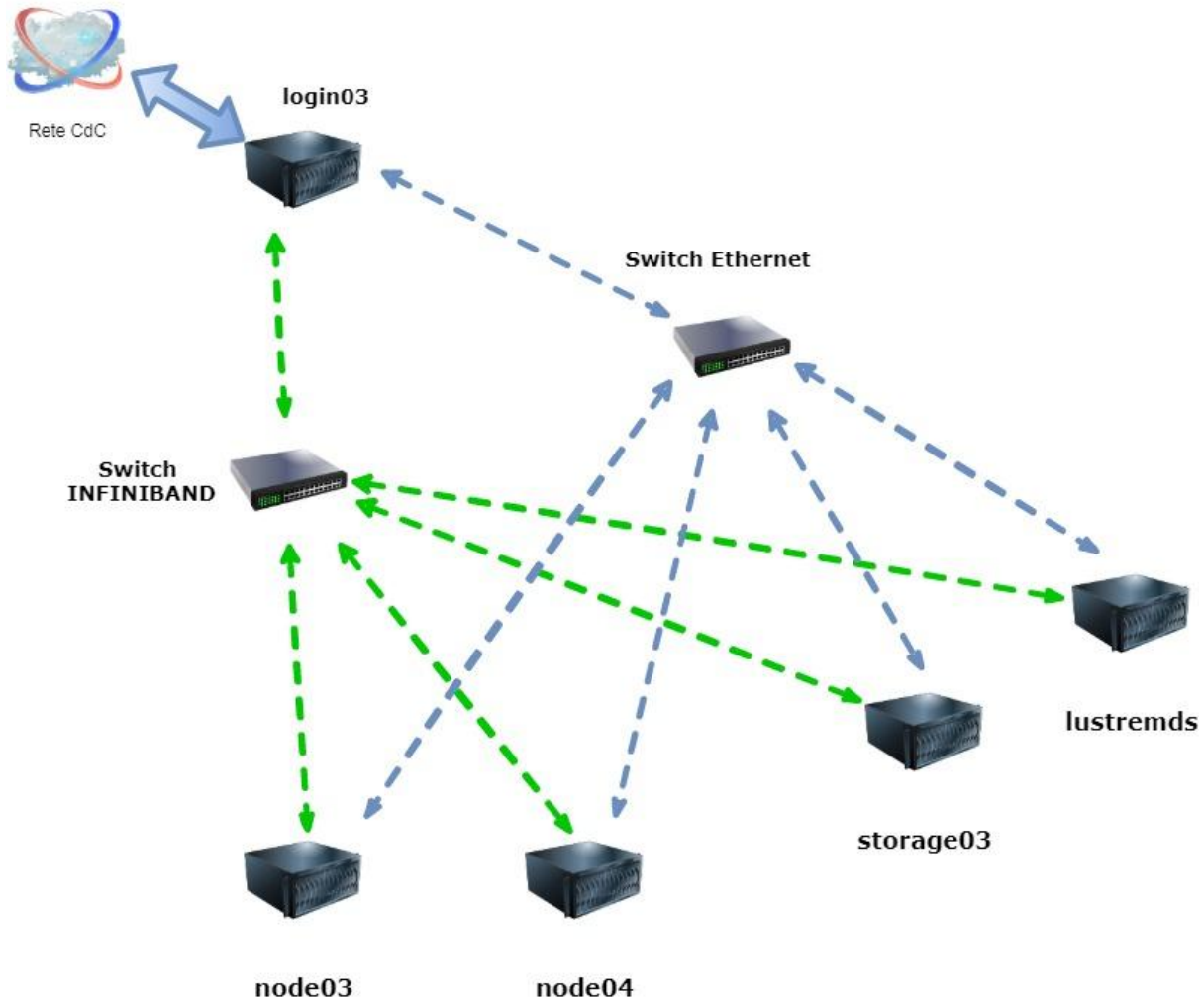
Il cluster di calcolo di OAS utilizzava IBM GPFS come filesystem condiviso e pur avendo delle prestazioni eccellenti, nonché un buon livello di sicurezza, aveva il difetto di essere molto costoso.
La licenza risaliva ad una versione datata del software in quanto IBM richiede una licenza annuale per poter utilizzare le versioni più recenti e ciò ne rendeva il mantenimento troppo oneroso.
Un aspetto negativo riscontrato con la versione in nostro possesso era lo sporadico calo delle prestazioni con conseguente blocco del sistema e la necessità di riavviare le macchine. Le motivazioni di tale fenomeno risiedevano probabilmente nella vetustà della versione di GPFS licenziata e nella disomogeneità dei sistemi operativi dei nodi che lo implementavano.
In effetti la maggior parte dei nodi implementavano, come Sistema Operativo, Centos 6 e questo comportava una forte limitazione nell'uso del software più aggiornato, ad esempio l'uso dei container era quasi impossibile.
C'era anche un problema di sicurezza, le vecchie versioni del Sistema Operativo non ricevono più gli aggiornamenti di sicurezza, che però era di lieve entità in quanto quasi tutti i nodi appartenevano ad una rete locale.

LUSTRE è un file system distribuito molto usato in ambito scientifico (se non il più usato), è gratuito ed è mantenuto dalla comunità degli utilizzatori.
È presente una grande mole di documentazione ed esempi che però non sono aggiornati in modo organico, bisogna fare molta attenzione alla versione di LUSTRE cui si riferiscono.
Inoltre la grande maggioranza delle implementazioni usano Ethernet come sistema di collegamento tra i nodi e sono molto poche le implementazioni, e quindi le relative documentazioni, che usano Infiniband.
L'utilizzo di Infiniband è stato dettato dall'hardware già presente nel cluster di calcolo precedente, nonostante la versione datata delle schede (e dello switch) Infiniband in nostro posesso si è deciso di continuare ad usarlo per via della bassa latenza che lo caratterizza.
Un altro problema legato all'utilizzo della tecnologia Infiniband è l'alto costo degli apparati, questo comporterà un attento ragionamento quando si presenterà l'occasione di espandere ed aggiornare il cluster.
Per l'implementazione di tutto il sistema si è fatto riferimento al manuale operativo "Lustre* Software Release 2.x" nella versione del 2017 rilasciato da Intel e nella pagina wiki: "https://wiki.lustre.org" fonte preziosa di informazioni soprattutto per la parte relativa ad Infiniband.
Si è seguita la strategia di pianificare attentamente la configurazione desiderata, ciò ha permesso di individuare le caratteristiche tecniche da implementare e soprattutto i comandi corretti da impartire velocizzando tutta la procedura e riducendone la complessità totale.
Sebbene Lustre preveda la possibilità di avere delle repliche a livello di file per aumentare la sicurezza si è scelto di non farne uso in quanto si sarebbe dovuto raddoppiare lo storage.
La strategia scelta è stata di lasciare l'intero filesystem come "scratch" ed usare un altro filesystem, molto più piccolo ma con backup, per conservare i dati.

2. Hardware scelto

La scelta effettuata è stata quella di utilizzare hardware già presente nel Centro di Calcolo per quanto possibile e di acquistare lo stretto necessario.
Si è deciso di estrapolare 2 nodi dal cluster di calcolo già presente ed una delle macchine usate per accedervi (nodi di login) in modo da replicare la struttura concettuale del vecchio cluster: ovvero dei nodi di calcolo e dei nodi di storage collegati da una sottorete dedicata fisicamente separata dalla rete

di struttura; e dei nodi di login che fanno da ponte e permettono di accedere alle risorse.
Di conseguenza i nodi “interni” possiedono degli indirizzi di rete che sono visibili solo tra di loro e comunicano tra loro tramite uno (o più) switch dedicati.
I nodi di frontiera (nodi di login) fanno parte sia della rete dedicata (e chiusa) sia della normale rete di struttura.
Vengono usati due switch che realizzano due reti geograficamente separate: uno switch Infiniband ed uno switch ethernet con porte RJ45.
La rete Infiniband, che ha un buon throughput ed una bassa latenza, viene impiegata per lo scambio di dati e metadati del file system condiviso cioè LUSTRE.
La rete ethernet viene invece impiegata per il resto del traffico tra i vari nodi.
Lo schema seguente raffigura come sono collegate tutte le macchine.



Schema 1

Trattandosi di sottoreti private per far comunicare i vari elementi tra loro non viene utilizzato il server DNS ma si ricorre al file /etc/hosts.
Nello schema 1 si notano le due macchine di nuova acquisizione, storage03 e lustremds.

La tabella 1 riassume le caratteristiche principali delle varie macchine.

Nome	Core	RAM
node03	64	194 GB
node04	64	270 GB
login02	8	32 GB
storage03	32	64 GB
lustremds	20	64 GB

3. Installazione Software

L'installazione del software sulle macchine che realizzano l'architettura di Lustre ha dei passaggi generali e comuni a tutte e dei passaggi caratteristici di ciascuna.
Per i passaggi comuni si è proceduto con l'installazione di CentOS 7.9 con i relativi aggiornamenti, la disabilitazione di Selinux e del firewall (tranne su login02 essendo un nodo di frontiera).
Disabilitare il firewall è giustificato dal fatto che le macchine usano due reti private fisicamente separate dal resto della rete e gli unici gateway sono i nodi di login.
Una operazione molto importante è stata la configurazione delle interfacce di rete.
Per prima si è configurata la rete Ethernet realizzando dove possibile un bonding[1] tra le porte fisiche ed assegnando manualmente un indirizzo ip sulla sottorete 192.169.210.x.

[1] Il bonding è una procedura in cui si realizza una porta di rete logica attraverso l'unione di due porte fisiche. In questo modo si raddoppia la banda disponibile e si introduce ridondanza.

Il bonding è stato realizzato su tutte le macchine tranne storage03 che utilizza una singola interfaccia a 10 Gbps, mentre le altre macchine hanno interfacce a 1 Gbps.
Successivamente si è configurata la rete Infiniband assegnando manualmente un indirizzo sulla sottorete 192.168.7.x.
La configurazione di Infiniband è stata molto difficoltosa sia perché la documentazione è scarsa e sia perché c'è una incompatibilità tra il modulo Infiniband del kernel modificato di Lustre e le schede di marca Intel.
Inizialmente la rete sembra andare ma quando si va a configurare Lustre si ricevono sempre messaggi di errore, si è scoperto che Lustre fa riferimento, all'interno dei moduli del suo kernel, alle librerie Mellanox.
Queste librerie non sono compatibili con i dispositivi Intel ed essendo la nostra dotazione un misto di schede Intel e Mellanox si è creata molta confusione.
La soluzione è stata quella di utilizzare solo schede Mellanox.
In linea teorica è possibile usare schede Intel sui Client in quanto questi ultimi non usano un kernel modificato ma si consiglia ugualmente di impiegare schede Mellanox.
Il protocollo Infiniband prevede che un nodo funga da master per tutta la rete, in questo nodo deve essere attivo il servizio rdma che è contenuto nel pacchetto rdma-core.
Come spiegato sopra viene fatto uso del file /etc/hosts per identificare gli indirizzi ip associati a tutte le macchine, per ottenere un risultato omogeneo si è copiato lo stesso file su tutti i nodi.
Successivamente ci si è accertati che tutte le macchine si vedessero tra loro su tutte le interfacce, dedicando particolare attenzione ad Infiniband per le ragioni di cui sopra.
Un altro aspetto da tenere presente è la coerenza nella misurazione del tempo inteso come data e ora.
Per ottenere questa coerenza si è attivato il servizio ntp[2] e lo si è configurato per sincronizzarsi con il server ntp1.inrim.it dell'Istituto Nazionale di Ricerca Metrologica.
In figura 2 si riporta il punto del file /etc/ntp.conf dove viene specificata questa configurazione.

[2] Network time Protocol

```
# For more information about this file, see the man pages
# ntp.conf(5), ntp_acc(5), ntp_auth(5), ntp_clock(5), ntp_misc(5), ntp_mon(5).

driftfile /var/lib/ntp/drift

# Permit time synchronization with our time source, but do not
# permit the source to query or modify the service on this system.
restrict default nomodify notrap nopeer noquery

# Permit all access over the loopback interface.  This could
# be tightened as well, but to do so would effect some of
# the administrative functions.
restrict 127.0.0.1
restrict ::1

# Hosts on local network are less restricted.
#restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap

# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
#server 0.centos.pool.ntp.org iburst
#server 1.centos.pool.ntp.org iburst
#server 2.centos.pool.ntp.org iburst
#server 3.centos.pool.ntp.org iburst


server ntp1.inrim.it iburst
server ntp2.inrim.it iburst

#broadcast 192.168.1.255 autokey        # broadcast server
#broadcastclient                        # broadcast client
#broadcast 224.0.1.1 autokey            # multicast server
#multicastclient 224.0.1.1              # multicast client
#manycastserver 239.255.254.254         # manycast server
#manycastclient 239.255.254.254 autokey # manycast client
```

Figura 1 ntp.conf

Installazione Repository

Per il processo di installazione di Lustre si è fatto uso della wiki https://wiki.lustre.org/Installing_the_Lustre_Software in cui si raccomanda di usare un repository locale.
Si è quindi creata una macchina virtuale con un servizio httpd e su cui si è creato un repository di Lustre.
Nei vari nodi poi all'atto di installare Lustre si è fatto riferimento a questo repository.
Di seguito si riportano i principali passaggi eseguiti.
– Si è creata una semplice macchina virtuale con un server httpd
– Si è creato il file lustre-repo.conf per importare i dati del repository

vi /tmp/lustre-repo.conf

– Si è scelta la versione più recente di Lustre stabile e con il supporto a Infiniband

```
[lustre-server]
name=lustre-server
#baseurl=https://downloads.whamcloud.com/public/lustre/lustre-2.12.6-ib/el7/server
baseurl=https://downloads.whamcloud.com/public/lustre/lustre-2.12.6-ib/MOFED-4.9-2.2.4.0/el7.9.2009/server
# exclude=*debuginfo*
gpgcheck=0

[lustre-client]
name=lustre-client
#baseurl=https://downloads.whamcloud.com/public/lustre/lustre-2.12.6-ib/el7/client
baseurl=https://downloads.whamcloud.com/public/lustre/lustre-2.12.6-ib/MOFED-4.9-2.2.4.0/el7.9.2009/client
# exclude=*debuginfo*
gpgcheck=0

[e2fsprogs-wc]
name=e2fsprogs-wc
baseurl=https://downloads.whamcloud.com/public/e2fsprogs/latest/el7
# exclude=*debuginfo*
gpgcheck=0
```

– Si è creato il repo e si sono scaricati i dati

```
mkdir -p /var/www/html/repo

cd /var/www/html/repo

reposync -c /tmp/lustre-repo.conf -n -r lustre-server -r lustre-client -r e2fsprogs-wc
```

– Si sono creati i metadati per il repository

```
for i in e2fsprogs-wc lustre-client lustre-server; do (cd $i && createrepo .); done
```

– Si è finalizzata la creazione del repository

```
cd lustre-server/
createrepo .
cd ..
cd lustre-client/
createrepo .
cd ..
cd e2fsprogs-wc/
```

– Si è creato in /var/www/html il file lustre.repo che i vari nodi useranno col comando yum per installare Lustre

```
[lustre-server]
name=lustre-server
baseurl=https://lustrerepo.hide.bo.iasf/repo/lustre-server
enabled=0
gpgcheck=0
proxy=_none_

[lustre-client]
name=lustre-client
baseurl=https://lustrerepo.hide.bo.iasf/repo/lustre-client
enabled=0
gpgcheck=0

[e2fsprogs-wc]
name=e2fsprogs-wc
baseurl=https://lustrerepo.hide.bo.iasf/repo/e2fsprogs-wc
enabled=0
gpgcheck=0
```

– Si è riavviato il server httpd per renderlo operativo
– Si è propagato il file lustre.repo su tutti nodi nella directory /etc/yum.repos.d/

4. Schema Lustre

I componenti di Lustre sono ben illustrati nella figura seguente reperita su https://wiki.lustre.org/Understanding_Lustre_Internals

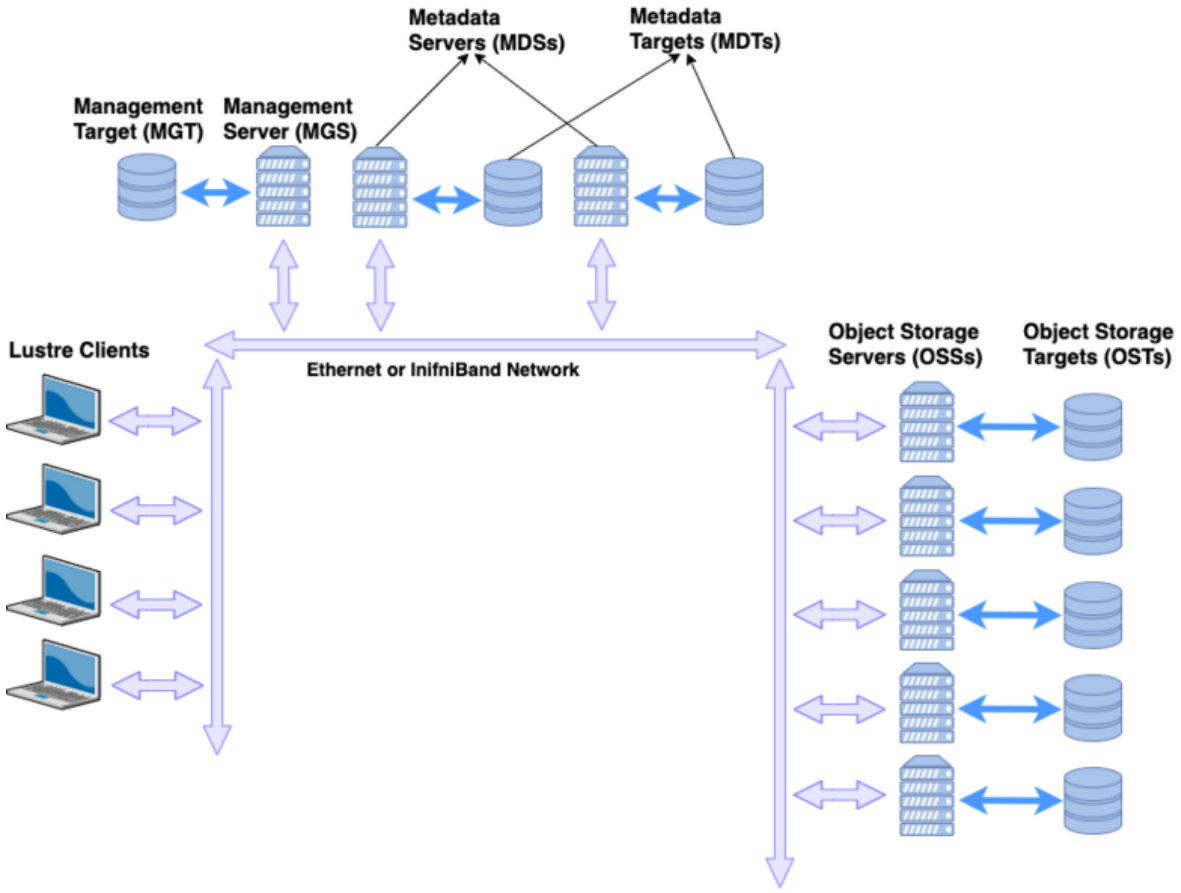


Figura 2 Schema Lustre

Come già detto Lustre è una piattaforma open source per un file system distribuito e parallelo i cui componenti principali sono:

MGS: Management Server, contiene la configurazione di Lustre ed un registro dei server e client attivi.
MGT: Management Target, è lo storage usato dal MGS.
MDS: Metadata Server, contiene i metadati del file system (fondamentalmente gli inodes) e gli fornisce il namespace.
MDT: Metadata Target, è lo storage usato dal MDS.
OSS: Object Storage Server, contiene lo storage in cui vengono scritti i file, questi possono essere scritti su stripe suddivise sui vari OST (con policy anche articolate). L'insieme dei vari OSS costituisce lo storage a disposizione e contribuisce al throughput complessivo del sistema.

OST: Object Storage Target, è l'unità di storage a blocchi usata dall'OSS e normalmente ne sono presenti più di uno per server. Può corrispondere ad una unità fisica ma spesso è un dispositivo logico formato da più unità fisiche (RAID).
LNET: Lustre Network, è il protocollo di rete specifico di Lustre in grado di aggregare il traffico su interfacce fisiche indipendenti e di supportare in modo eterogeneo diverse tecnologie di rete (Ethernet, Infiniband, Omni Path). Tutte le transazioni di IO[3] del file system passano attraverso la rete ed i client possono anche non avere storage locale (a parte per l'installazione dell'OS[4]).
Client: montano il file system Lustre attraverso il protocollo di rete LNET e l'OS lo vede come un normale file system di tipo POSIX[5].

Di MDS+MDT ce ne possono essere diversi per fornire ridondanza o per far fronte ad un aumento dello storage e del relativo ingrandimento del file system.

5. Metadata Server/Management Server

Questo server è il cervello del sistema Lustre, le due funzioni, Metadata Server (MDS) e Management Server (MGT), dovrebbero essere implementate su macchine separate ma spesso si trovano riunite. Questo perché non c'è necessità di prestazioni computazionali elevate né di molta memoria.
Il più importante è l'MDS, esso è composto dal server MDS stesso e dal suo storage MDT (Meta Data Target), così come l'MGS è composto da se stesso e dall'MDT.
L'MDS è cruciale in quanto se si perdono i metadati si perde tutto il file system senza possibilità di recupero. L'unica pratica per mitigare questa situazione è effettuare un backup dei dati.
A causa delle risorse limitate si è utilizzato un solo MDS ma si è cercato di minimizzare il rischio di rotture.
Si è acquistata una macchina nuova con doppio alimentatore e si sono utilizzati dischi ssd configurati in RAID 1.
La macchina è stata chiamata lustremds.
Per aggiungere un grado di sicurezza in più l'MDT è stato creato come LVM[6], questo ci dà la possibilità di creare uno snapshot del volume logico.

[3] Input Output
[4] Operating System
[5] **P**ortable **O**perating **S**ystem **I**nterface for Uni**x**
[6] Logical Volume Manager

```
[root@lustremds ~]# lsblk
NAME                MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda                   8:0    0 223,5G  0 disk
├─sda1                8:1    0     1G  0 part /boot
├─sda2                8:2    0     8G  0 part [SWAP]
└─sda3                8:3    0 214,5G  0 part /
sdb                   8:16   0 894,1G  0 disk
└─sdb1                8:17   0 894,1G  0 part
  └─lustremds-MDT0 253:0    0   894G  0 lvm  /mnt/mdt
sdc                   8:32   0 237,5G  0 disk
└─sdc1                8:33   0     1G  0 part /mnt/mgt
sdd                   8:48   0 931,3G  0 disk
└─sdd1                8:49   0 931,3G  0 part
```

Fig.3 struttura dei dischi di lustremds

In figura 3 si può notare che il volume di storage dedicato ad MDS è di tipo lvm ed è montato in /mnt/mdt, mentre il volume do storage dedicato ad MGS è un normale volume montato su /mnt/mgt.
Il terzo volume (sdd1) è il volume logico dedicato allo snapshot, non si vede perché non è ancora stato formattato e montato.
Una volta finiti i passi descritti nel paragrafo precedente si ha la possibilità di installare Lustre.
Si scarica il file repo del nostro server lustre repository

```
cd /etc/yum.repos.d/
wget --no-check-certificate   https://lustrerepo.hide.bo.iasf/lustre.repo
```

Si installano alcuni pacchetti necessari

```
yum install asciidoc audit-libs-devel automake bc
yum install binutils-devel bison device-mapper-devel elfutils-devel
yum install elfutils-libelf-devel expect flex gcc gcc-c++ git glib2 glib2-devel
hmaccalc keyutils-libs-devel krb5-devel ksh libattr-devel libblkid-devel
libselinux-devel libtool libuuid-devel libyaml-devel lsscsi make ncurses-devel
yum install net-snmp-devel net-tools newt-devel numactl-devel parted
patchutils pciutils-devel perl-ExtUtils-Embed pesign python-devel
redhat-rpm-config rpm-build systemd-devel tcl tcl-devel tk tk-devel wget
xmlto yum-utils zlib-devel
```

Si installa Lustre dal repo

```
yum --nogpgcheck --disablerepo=* --enablerepo=e2fsprogs-wc install
e2fsprogs

yum --nogpgcheck --disablerepo=base,extras,updates
--enablerepo=lustre-server install kernel kernel-devel kernel-headers
kernel-tools kernel-tools-libs kernel-tools-libs-devel

yum --nogpgcheck --enablerepo=lustre-server install kmod-lustre
kmod-lustre-osd-ldiskfs lustre-osd-ldiskfs-mount lustre
lustre-resource-agents

modprobe -v lustre
lustre_rmmod
```

Si formattano e si montano i volumi dedicati ad MDT e MDS

```
mkfs.lustre --mgs /dev/sdc1
mkfs.lustre --fsname=blasco --mgsnode=lustremds@tcp0,lustremds@o2ib0
--mdt --index=0 /dev/sdb1
mkdir /mnt/mgt
mkdir /mnt/mdt
mount -t lustre /dev/sdc1 /mnt/mgt
mount -t lustre /dev/sdb1 /mnt/mdt
```

Da notare come con la creazione dell'MDT si crea anche il filesystem e si deve assegnargli un nome.
Resta solo da configurare ed attivare LNET.
Si sono configurate sia le reti Infiniband che Ethernet, anche se viene usata solo la prima per lo scambio dei dati, e si è scelta una configurazione dinamica.
In questo modo si possono facilmente apportare correzioni o cambiamenti, ad esempio è facile passare da Infiniband ad Ethernet, ma si introduce un maggiore lavoro di gestione.
In teoria andrebbe prima attivata LNET e poi montati i volumi MDT ed MGT ma non ci sono conseguenze ad invertire l'ordine.

```
modprobe -v ko2iblnd
modprobe -v lnet
lnetctl lnet configure
lnetctl net show --verbose
lnetctl net add --net o2ib0 --if ib0
```

```
lnetctl net add --net tcp0 --if bond0
```

Se potrebbe utilizzare un unico disco per MDT e MGT in quanto lo spazio occupato da MDT è esiguo (~100MB), tuttavia se si vuole avere più di un filesystem è necessario averli separati (ci vuole un MDT per ogni filesystem).

6. Object Storage Server

Se l'MDS è il cervello di Lustre l'OSS ne è il cuore.
In una implementazione tipica prevede diversi OSS ciascuno con diversi OST, ciascun OST è identificato univocamente in tutto il sistema.
Con un corretto bilanciamento sul numero di OST si può ottenere un alto throughput aggregato che è una delle caratteristiche salienti di Lustre.
Attraverso un raid hardware si sono realizzati 3 volumi logici in RAID 6, ciascuno di 32,8 TB, che andranno a costituire gli OST.
La figura 4 mostra l'organizzazione dello storage del nostro server OSS che abbiamo chiamato storage03.

```
[root@storage03 ~]# lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0  32,8T  0 disk /mnt/ost0
sdb      8:16   0  32,8T  0 disk /mnt/ost1
sdc      8:32   0  32,8T  0 disk /mnt/ost2
sdd      8:48   0 238,4G  0 disk
├─sdd1   8:49   0   953M  0 part /boot
├─sdd2   8:50   0   230G  0 part /
└─sdd3   8:51   0   7,5G  0 part [SWAP]
```

Fig.4 storage03

L'installazione è simile a quella dell'MDS.
Si scarica il file repo

```
cd /etc/yum.repos.d/
wget --no-check-certificate   https://lustrerepo.hide.bo.iasf/lustre.repo
```

Si installano i pacchetti necessari

```
yum install asciidoc audit-libs-devel automake bc
yum install binutils-devel bison device-mapper-devel elfutils-devel
```

```
yum install elfutils-libelf-devel expect flex gcc gcc-c++ git glib2 glib2-devel
hmaccalc keyutils-libs-devel krb5-devel ksh libattr-devel libblkid-devel
libselinux-devel libtool libuuid-devel libyaml-devel lsscsi make ncurses-devel
yum install net-snmp-devel net-tools newt-devel numactl-devel parted
patchutils pciutils-devel perl-ExtUtils-Embed pesign python-devel
redhat-rpm-config rpm-build systemd-devel tcl tcl-devel tk tk-devel wget
xmlto yum-utils zlib-devel
```

Si installa Lustre

```
yum --nogpgcheck --disablerepo=* --enablerepo=e2fsprogs-wc install
e2fsprogs

yum --nogpgcheck --disablerepo=base,extras,updates
--enablerepo=lustre-server install kernel kernel-devel kernel-headers
kernel-tools kernel-tools-libs kernel-tools-libs-devel

yum --nogpgcheck --enablerepo=lustre-server install kmod-lustre
kmod-lustre-osd-ldiskfs lustre-osd-ldiskfs-mount lustre
lustre-resource-agents

modprobe -v lustre
lustre_rmmod
```

Si avvia LNET

```
modprobe -v ko2iblnd
modprobe -v lnet
lnetctl lnet configure
lnetctl net show --verbose
lnetctl net add --net o2ib0 --if ib0
lnetctl net add --net tcp0 --if eno1
```

Infine si creano e si montano i volumi storage

```
mkfs.lustre --ost --fsname=blasco --index=0
--mgsnode=192.168.7.60@o2ib0 /dev/sda

mkfs.lustre --ost --fsname=blasco --index=1
--mgsnode=192.168.7.60@o2ib0 /dev/sdb

mkfs.lustre --ost --fsname=blasco --index=2
```

```
--mgsnode=192.168.7.60@o2ib0 /dev/sdc
mkdir /mnt/ost0
mkdir /mnt/ost1
mkdir /mnt/ost2
mount -t lustre /dev/sda /mnt/ost0
mount -t lustre /dev/sdb /mnt/ost1
mount -t lustre /dev/sdc /mnt/ost2
```

Purtroppo durante il periodo di sperimentazione è successo che si sono esauriti gli inode nonostante uno storage complessivo di quasi 100 TB (sono stati scritti miliardi di piccoli file).
Questo ha comportato il blocco totale del filesystem.
Si è deciso di riformattare tutto lo storage, perdendo tutti i dati ma essendo un sistema sperimentale era stato messo in conto, aumentando il numero di inode per ogni OST.

```
mkfs.lustre --mkfsoptions="-i 204800" --reformat --ost --fsname=blasco
--index=0 --mgsnode=192.168.7.60@o2ib0 /dev/sda

mkfs.lustre --mkfsoptions="-i 204800" --reformat --ost --fsname=blasco
--index=1 --mgsnode=192.168.7.60@o2ib0 /dev/sdb

mkfs.lustre --mkfsoptions="-i 204800" --reformat --ost --fsname=blasco
--index=3 --mgsnode=192.168.7.60@o2ib0 /dev/sdc
```

7. Installazione Client

Per i client si segue una procedura molto simile ai server ma molto più semplice in quanto non è necessario usare un kernel modificato.
Nel nostro sistema i client corrispondono alle macchine: login02, node03, node04.
Dopo essersi assicurati che le schede di rete siano state configurate a dovere è sufficiente importare il file di configurazione del repository locale di Lustre e procedere all'installazione.

```
cd /etc/yum.repos.d/
wget --no-check-certificate   https://lustrerepo.hide.bo.iasf/lustre.repo
```

L’installazione è molto semplice

```
yum --nogpgcheck --enablerepo=lustre-server install kmod-lustre-client
lustre-client
modprobe -v lustre
lustre_rmmod
```

Si avvia il modulo LNET

```
modprobe -v ko2iblnd
modprobe -v lnet
lnetctl lnet configure
lnetctl net show --verbose
lnetctl net add --net o2ib0 --if ib0
lnetctl net add --net tcp0 --if em2
```

E si monta il filesystem Lustre

```
mount -t lustre 192.168.7.60@o2ib0:/blasco /blasco
```

La figura 5 mostra il filesystem montato sul client login02

```
[root@login02 ~]# df -ht lustre
File system                Dim. Usati Dispon. Uso% Montato su
192.168.7.60@o2ib:/blasco  130T   97T     27T  79% /blasco
```

Fig.5

8. Procedure di accensione e spegnimento

Visto e considerato che si è scelto un approccio dinamico nella configurazione del filesystem e del modulo LNET risulta molto utile avere una procedura corretta per l’accensione e lo spegnimento.
Si farà riferimento alle macchine che sono state usate per questo sistema.

<u>Spegnimento</u>

- ➢ Su ogni client smontare il filesystem

```
umount /blasco
```

➢ Sull’MDS smontare i vari MDS ed MDT in quest’ordine

```
umount /mnt/mdt
umount /mnt/mgt
```

➢ Sull’OSS smontare i vari OST, per storage03

```
umount /mnt/ost0
umount /mnt/ost1
umount /mnt/ost2
```

➢ Disabilitare il modulo LNET su tutte le macchine

```
lnetctl lnet unconfigure
```

Accensione

➢ Sull’MDS far partire il modulo LNET e poi in ordine montare l’MGS e l’MDT

```
modprobe -v ko2iblnd
modprobe -v lnet
lnetctl lnet configure
lnetctl net show --verbose
lnetctl net add --net o2ib0 --if ib0
lnetctl net add --net tcp0 --if bond0

mount -t lustre /dev/sdc1 /mnt/mgt
mount -t lustre /dev/sdb1 /mnt/mdt
```

➢ Sull’OSS far partire il modulo LNET e successivamente montare i vari OST

```
modprobe -v ko2iblnd
modprobe -v lnet
```

```
lnetctl lnet configure
lnetctl net show --verbose
lnetctl net add --net o2ib0 --if ib0
lnetctl net add --net tcp0 --if eno1
mount -t lustre /dev/sda /mnt/ost0
mount -t lustre /dev/sdb /mnt/ost1
mount -t lustre /dev/sdc /mnt/ost2
```

➢ Sui Client far partire il modulo LNET e montare il filesystem Lustre

```
modprobe -v ko2iblnd
modprobe -v lnet
lnetctl lnet configure
lnetctl net show --verbose
lnetctl net add --net o2ib0 --if ib0
lnetctl net add --net tcp0 --if em2

mount -t lustre 192.168.7.60@o2ib0:/blasco /blasco
```

9. Conclusioni

Non sono state valutate in maniera precisa le prestazioni ma sono state raccolte le impressioni d'uso da parte degli utenti ed il risultato è stato buono in quanto non c'è stata neanche una lamentela su eventuali lentezze del sistema, l'unico "inciampo" si è avuto col filesystem pieno quasi al 100% ed in quella situazione qualunque filesystem smette di funzionare.
Il sistema si è rivelato robusto in casi spegnimenti improvvisi e non ci sono state perdite di dati.