CREAZIONE CLUSTER OPENMOSIX FOR DUMMIES

Lo scopo di questo scritto e' la tracciatura passo-passo delle operazioni necessarie alla creazione di un cluster openMosix, da adesso in poi indicato con i termini 'cluster' o 'pool'; i computer costituenti il cluster saranno chiamati 'nodi'.

A CHI SI RIVOLGE QUESTO DOCUMENTO

Mi rivolgo principalmente a persone interessate alla creazione di un cluster di calcolo openMosix ma che non conoscono in modi approfondito l'argomento, percio' cerchero' di essere il piu' dettagliato possibile nell'esposizione.

Non ho la pretesa di spiegare per filo e per segno la storia di Mosix prima e openMosix poi, per queste informazioni, mi limito a rimandare il lettore allo "openMosix-HOWTO" (http://openmosix.sourceforge.net) ed alla documentazione reperibile in rete.

COSA E' OPENMOSIX?

openMosix e' la versione GPL di Mosix: col tempo si e' evoluto diventando un progetto a se' stante, replicando, migliorando ed espandendo le caratteristiche di Mosix.
OpenMosix e' una patch del kernel linux, che lo mette in grado di far migrare i processi da un nodo ad un altro, accellerandone il completamento.

COSA E' UN CLUSTER?

La definizione piu' generale di cluster é un insieme componenti hardware e software dedicati all'ottenimento di:

- servizi
- prestazioni
- gradi di affidabilita'

non ottenibili dalle singole unita' dell'insieme.

Qui prenderemo in esame le PRESTAZIONI, verra' cioe' creato un **CLUSTER DI CALCOLO.**

Basandosi sul concetto che "l'unione fa la forza" due o piu' PC correttamente configurati eseguiranno una serie di compiti in un tempo minore rispetto al singolo PC.

PREMESSA

In giro per la Rete si possono trovare HOW-TO piu' o meno chiari, in italiano o in inglese, ma sono sempre dedicati a RedHat o altre distribuzioni. Se cercate bene, troverete anche il mio per DEBIAN in italiano, (eh eh).

Questo documento e' una semplificazione ed un aggiornamento di un mio documento precedente, dove si spiegava come installare openMosix su una linuxbox DEBIAN/Linux: qui troverete come installare openMosix **INDIPENDENTEMENTE**
dalla distribuzione da voi usata. Ciononostante, ho usato debian per installare le linuxbox che trasformero' in nodi del cluster.

PREREQUISITI

si presuppone che:
- si usino PC con delle CPU x86 compatibili (niente Machintosh/Apple). Se avete una cpu non x86, potete usare un emulatore, come **qemu** si veda per informazioni http://fabrice.bellard.free.fr/qemu/
- se installate i pc da zero, eseguiate una installazione minimale, sufficente a trovarvi in riga di comando, e basta, niente grafica o altro. Per darvi una idea, una installazione minimale di Debian/woody occupa meno di 120M.
- disponiate di almeno due pc con installato linux
- i pc siano dotati di scheda di rete (funzionante!)
- i pc abbiano degli IP fissi.
- che il lettore abbia una minima competenza della riga di comando BASH, poiche' faremo TUTTO il lavoro da consoll, (sui pc la grafica non e' nemmeno installata: comunque, nulla vi vieta di lavorare in modalita' grafica).

Tuttavia, poiche' lo scopo del documento e' di avvicinare il maggior numero possibile di persone al meraviglioso mondo del clustering, cercero' di spiegare le cose nel modo piu' dettagliato possibile, (usando la riga di comado, che e' a disposizione di tutti), in modo che openMosix possa essere installato anche dall'ultimo degli imbecilli. (Se pensi che stia riferendomi a te, molto probabilmente e' cosi' ;-)).

ALCUNI DATI TENCNICI

Ovviamente, l'hardware minimo necessario sono ALMENO due computer (desktop o portatili), entrambi dotati di una scheda di rete. Comunque, prevederemo un cluster composto da un massimo di otto nodi, tutti con ip fisso, da 192.168.1.10 a 192.168.1.17: il nodo master avra' indirizzo 192.168.1.10. I nodi dovranno essere connessi tramite hub o switch: se disponete di due soli nodi, sara' sufficente un cavo di rete INCROCIATO e avrete un cluster minimale.

SOFTWARE NECESSARIO

Servono i sorgenti PULITI, cioe' senza patch. I sorgenti del kernel delle varie distribuzioni (Debian, slackware, etc) **NON** vanno bene, occorre procurarsi i sorgenti originali. Il sito da cui scaricare e' (http://www.kernel.org/pub/linux/kernel/v2.4/).
Il kernel che verra' usato e' l'ultimo supportato da openMosix, cioé il 2.4.26. Il file si chiama `linux-2.4.26.tar.bz2`.
La patch per abilitare il kernel al clustering e' `openMosix-2.4.26-1.bz2` reperibile partendo dal sito (http://openmosix.sourceforge.net/#LatestRealase) ; sempre nello stesso sito, procuratevi le utility per amministrare il cluster: l'ultima versione del file si chiama `openMosix-tools-0.3.6-2.tar.gz`

Tutte le operazioni vanno eseguite da root, cioe' con i diritti di amministratore (il # all'inizo riga indica che si sta lavorando con i diritti di root)

PREPARIAMO I NODI

Parto dal presupposto che abbiate un pc con linux installato: se cosi' non fosse e volete installare da zero linux su un pc da convertire in un nodo, fate riferimento ad uno dei tanti howto che trattano l'installazione di linux: se ritenete possano esservi utili, potete leggere i miei howto sull'installazione di Debian che trovate su www.newopenbrains.org/roberto o www.scomodo.org/~roberto. Usando i miei howto, ed il primo CD di Debian/woody, sarete in grado di approntare una linuxbox.

Ora che avete installato almeno due linuxbox minimali, proseguiamo.

Ovviamente, ci sono dei pacchetti che devono essere presenti sul pc per poter lavorare: ecco la lista.
- mc
- make

- cpp-3.x
- gcc-3.x
- patch
- libncurses5-dev
- bzip2

Per quanto mi riguarda, la "x" vale 0 (non 'o' maiuscola, ma zero) poiche' al momento ho a disposizione la versione 3.0 dei software citati (uso Debian Woody). Puo' darsi che questi pacchetti siano gia' installati, magari ad una versione diversa per quanto riguarda i compilatori. Mi sento di consigliare per il cpp e gcc ALMENO la versione 3.0 o superiore.
Usate il vostro tool di gestione pacchetti per installarli: Per DEBIAN, il comando e' (da consoll)

```
# apt-get install nome_pacchetto
```

Ho consigliato di installare mc: se qualcuno di ricorda il Norton Commander(TM) per dos, ecco, mc (Midnight Commander) e' il suo clone per linux. Si tratta di un file manager testuale, che integra anche un programma di editazione testi, mcedit. Con mc potrete gestire il vostro pc (copiare, spostare, editare files, etc). Se volete elaborare dei file di testo senza avviare mc, e' sufficiente digitare

```
mcedit percorso/nomefile
```

Nulla vieta di usare il "terribile" vi, (si, si chiama proprio vi, "v" "i") presente di default su TUTTE le distribuzioni linux, oppure installatene uno a vostra scelta, tipo nano (niente a che fare con circhi o ballerine) oppure pico (niente a che fare con Paperino o Zio Paperone).

Adesso, dobbiamo dire al pc di usare i compilatori appena intallati:

```
# cd /usr/bin
# rm -f cpp ; ln -s cpp-3.x cpp
# rm -f gcc ; ln -s gcc-3.x gcc
```

E FINALMENTE SI COMINCIA

Gli svilupatori di openMosix numerano le loro patch con la stessa numerazione di linux: per cui, la patch openMosix versione 2.4.26 dovra' essere applicata a linux versione 2.4.26. Non usate versioni diverse di linux e openMosix perche NON funzioneranno.

Bene, ammesso che vi siete procurati il kernel, la patch openmosix e gli openMosix-tools, procediamo.

E' OVVIO che TUTTI i nodi dovranno subire le modifiche (ricompilazione kernel, compilazione openMosix-tools, configurazione scheda di rete) per poter diventare parte del cluster.

Indico qui di seguito i passi necessari a ricompilare il kernel e modificare il lilo.

Ora daro' di seguito tutta la lista di comandi per creare il kernel:

```
# mv /directory_di_partenza/linux-2.4.26.tar.bz2 /usr/src/
# mv /directory_di_partenza/openMosix-2.4.26-1.bz2 /usr/src/
# cd /usr/src
# tar jxvf linux-2.4.26.tar.bz2
# rm -f linux
# ln -s linux-2.4.26 linux
# cd linux
# bzcat /usr/src/openMosix-2.4.26-1.bz2 | patch -p1
# make menuconfig
```

Ora si presenta la scelta dei parametri di openMosix, configurare le opzioni come qui indicato:

```
[*] openMosix process migration support
[ ] Support clusters with a complex network topology
```

```
[*] Stricter security on openMosix ports
(3) Level of process-identity disclosure (0-3)
[ ] openMosix File-System
[ ] Poll/Select exceptions on pipes
[ ] Disable OOM Killer
```

NOTA: non ho abilitato l'opzione openMosix filesystem per motivi di sicurezza, ma piu' avanti nel documento parlo come se lo avessi fatto. In poche parole, l'openmosix filesystem (omfs) permette di vedere il filesystem di tutti i nodi partendo da /, e con tutti i permessi di lettura/scrittura.

NOTA2: ATTENZIONE! le opzioni di configurazione kernel, per quanto riguarda la parte openMosix, DEVONO essere identiche su TUTTI i nodi. Cioe', queste opzioni qui sopra DEVONO essere identiche su tutti i nodi del cluster. Ovvimente, se avete un pc con un disco SCSI, per quanto riguarda quel pc potrete attivare il sopposto SCSI.

NOTA3: i nodi DEVONO montare tutti la stessa versione di kernel+openMosix. se hai due nodi, entrambi devono avere il 2.4.22 o (per esempio) il 2.4.20, ma il clustr NON FUNZIONA se eseguite su uno il 2.4.22 e sull'altro il 2.4.20.

Configurate il resto del kernel: vi consiglio solo di abilitare la scheda di rete e niente altro, tipo usb, scheda audio, etc, tanto per tenere il kernel piccolo: DOPO che vedrete il kernel funzionante con openMosix, potrete aggiuntere altra roba, per il momento, limitiamoci a fare funzionare il cluster. Finite la parametrizzazione, salvate ed uscite.

IL TERRORE NEI VOSTRI OCCHI

ALT! PANICO!! parametrizzare il resto del kernel?!?! brivido, terrore, raccapriccio!! e come si fa?!? Cosa metto? Cosa tolgo? paura, paura!!!
tranquillo, hombre!
Qui, tutto per te, c'e' un bel .config gia' pronto: si tratta di una configurazione del kernel per cpu Pentium MMX o superiore. La configurazione e' minimale: niente, audio, usb, SCSI, etc, solo openmosix e schede di rete, ANCHE per portatili.

Se non sai come parametrizzare il resto del kernel, da bravo: prendi il file .config (si, il puntino all'inizio ci DEVE essere) e copialo all'interno della directory /usr/src/linux e passa DIRETTAMENTE alla compilazione.

Ora occorre compilare il kernel:

```
# make dep
# make bzImage
# make modules
# make modules_install
# depmod -a
# cp /usr/src/linux/arch/i386/boot/bzImage /boot/linux-2.4.26-oM
```

NOTA: io ho previsto il kernel monolitico, cioe' senza moduli: comunque, per completezza , ho aggiunto anche make modules e make modules_install e depmod -a, che servono solo per il kernel modulare, nel caso voi scegliate quest'ultimo in fase di parametrizzazione.

Il kernel é compilato e messo nella directory /boot, adesso occorre modificare il /etc/lilo.conf; editatelo col vostro text editor preferito) ed assicuratevi che contenga una riga con la parola:

```
prompt
```

senza il # davanti. Inoltre il file deve contenere queste righe, per abilitare il kernel ricompilato:

```
image=/boot/linux-2.4.26-oM
label=linux-2.4.26-oM
read-only
optional
```

Salvate il file e tornate a riga di comando e verificate che lilo.conf non contenga errori con comando:

```
# lilo -t
```

Se vedete avvisi di warning o errori, correggete il lilo.conf: altrimenti, abilitate le modifiche fatte col comando:

```
# lilo -v
```

RICOMPILAZIONE COMANDI

Non basta modificare il kernel per openMosix, occorre anche compilare i comandi dedicati. Esistono dei comandi senza i quali il cluster non può essere configurato, monitorato ed amministrato. Qui di seguito le operazioni per crearli dai sorgenti.

```
# mv /directory_di_partenza/openmosix-tools-0.3.6.tar.bz2 /usr/src/
# cd /usr/src
# tar zxvf openmosix-tools-0.3.6-2.tar.gz
# cd openmosix-tools-0.3.6-2
```

Durante la compilazione dei comandi, gli script si aspettano di trovare i sorgenti del kernel in un posto ben preciso, per cui creiamo un link simbolico:

```
# ln -s /usr/src/linux /usr/src/linux-openmosix
```

La lettura del file README presente in /etc/usr/openmosix-tools-0.3.6-2 e' chiarificatrice di come compilare i tools. Comunque, questi tre comandi saranno sufficienti nella maggioranza dei casi. Adesso, lanciamo la configurazione, compilazione ed installazione dei comandi:

```
# ./configure
# make
# make install
```

Se durante la configurazione si ferma segnalando un errore, probabilmente non avete installato qualche cosa: guardate il messaggio di errore e tentate di correggere. Se invece tutto e' andato bene, a questo punto i comandi sono stati installati e anche gli script di avvio sono stati configurati.

CONFIGURAZIONE SCHEDA DI RETE

Per quei pochi che ancora non lo sanno, la configurazione della scheda di rete avviene grazie al file:

```
/etc/network/interfaces
```

Ora, questo file configura le schede di rete e la scheda (virtuale) di loopback. Di solito la scheda di rete e' una sola (eth0).
Per chiarezza, allego il contenuto di /etc/network/interfaces che e' presente sul mio nodo master:

```
#questa e' l'interfaccia di loopback
auto lo
iface lo inet loopback
```

```
#questa e' l'interfaccia di rete
auto eth0
iface eth0 inet static
    address 192.168.1.10
    netmask 255.255.255.0
    network 192.168.1.0
    broadcast 192.168.1.255
```

Ovviamente, l'indirizzo 192.168.1.10 E' VALIDO SOLO PER IL NODO MASTER, sugli altri nodi sara' rispettivamente 192.168.1.11, 192.168.1.12, etc.

Ora configurate il file /etc/hosts, inserendo l'ip del nodo: se il file /etc/hosts contiene una riga tipo:

```
127.0.0.1 master localhost
```

Occorre modificare il file nel seguente modo: (dovrete eseguire le modifiche anche sugli altri nodi, con le dovute differenze di nome/indirizzo IP)

```
198.168.1.10 master
127.0.0.1 localhost
```

PREPARIAMO I NODI

```
# mkdir /mfs
```

Questo comando crea una directory, all'interno della quale verranno automaticamente create dal driver oM delle sotto-directory /1, /2, /3 etc, tante quante i nodi presenti nel cluster.

editate il file /etc/fstab ed assicuratevi che contenga questa riga, che permettera' il montaggio del file-system oM, nel caso in cui sia abilitato in /etc/openmosix/openmosix.config (si veda oltre) e a patto che sia stata prevista tale opzione durante la ricompilazione del kernel.

```
mfs /mfs mfs odfsa=1 0 0
```

Ultimo passo:
editate il file /etc/openmosix.map ed assicuratevi che contenga gli indirizzi dei vari nodi: per comodità lo allego integralmente.

```
1  192.168.1.10  1
2  192.168.1.11  1
3  192.168.1.12  1
4  192.168.1.13  1
5  192.168.1.14  1
6  192.168.1.15  1
7  192.168.1.16  1
8  192.168.1.17  1
```

Adesso fate uno shutdown ed al reboot scegliete linux-2.4.26-oM

PARTENZA CLUSTER

A questo punto, se non é attiva, attivate la scheda di rete con:

```
# /etc/init.d/networking restart
```

Se per qualche motivo non e' partito lo script di attivazione openMosix attivatelo con:

```
# /etc/init.d/openmosix restart
```

Bene! Potete rendere il kernel con opemosix quello che parte di default, modificando il file /etc/lilo.conf.

Per verificare il funzionamento del nodo, (sto lavorando sul master) digitate in comando:

```
# /etc/init.d/openmosix status
```

risposta che ottengo é la seguente: (sto lavorando sul master)

```
This is OpenMosix node #1
Network protocol: 2 (AF_INET)
OpenMosix range     1-1     begins at master
OpenMosix range     2-2     begins at 192.168.1.11
OpenMosix range     3-3     begins at 192.168.1.12
OpenMosix range     4-4     begins at 192.168.1.13
OpenMosix range     5-5     begins at 192.168.1.14
OpenMosix range     6-6     begins at 192.168.1.15
OpenMosix range     7-7     begins at 192.168.1.16
OpenMosix range     8-8     begins at 192.168.1.17
Total configured: 8
```

Notate il fatto che oM viene attivato PRIMA che ci si faccia il logon sul sistema, quindi la potenza di calcolo dei nodi sarà disponibile appena accesi, poiché oM funziona come se fosse un qualsiasi processo che gira in background.

AFFINIAMO IL CLUSTER

Editando il file /etc/openmosix/openmosix.config, e' possibile modificare la configurazione che openMosix utilizza di default. Analizzerenmo solo qualche voce, che e' interessante o che ha qualche attinenza col nostro cluster, che, lo ricordiamo, e' composto da un massimo di otto nodi con IP fisso.

AUTODISC=1 Questa opzione permette di attivare il programma omdiscd, che si occupa di interrogare il network, alla ricerca di altri nodi del cluster. Questa opzione e' molto comoda nel caso in cui il file /etc/openmosix.map sia danneggiato o quando di collegano nuovi nodi e non si ha la possibilita' di aggiornare il file /etc/openmosix.map presente su tutti gli altri nodi oppure quando i nodi non hanno IP fissi ma dinamici, forniti da un server DHCP. Questa opzione e' di default disabilitata.

AUTODISCIF=eth1 Questa opzione permette di dire al programma omdiscd quale scheda di rete usare per collegarsi agli altri nodi. E' possibile attivare da eth0 a eth5. Questa opzione e' disabilitata poiche' stiamo usando una sola interfaccia di rete, la solita eth0.

MOSGATES=numero Indica quanti livelli di gateways sono tra un nodo e l'altro: questa opzione si attiva nel caso in cui il cluster preveda un alto numero di nodi, suddivisi in sotto-gruppi, collegati attraverso gateway. Non e' il nostro caso.

MIGRATE=yes Questa opzione premette ai jobs lanciati su questo nodo di migrare (per essere elaborati) su altri nodi. Questa opzione e' impostata a "yes" di default. Se la impostate a "no", questo nodo non richiedera' aiuto agli altri per processare i propri jobs.

BLOCK=no l'ozione BLOCK e' l'opposto di MIGRATE. Questa opzione premette ai jobs lanciati su altri nodi di migrare (per essere elaborati) su questo nodo. Questa opzione e' impostata a "no" di default. Se la impostate a "yes", questo nodo non aiutera' gli altri nodi ad elaborare i loro jobs.

MFS=yes Questo parametro (qualora voi abbiate abilitato il supporto all'openMosix filesystem durante la parametrizzazione del kernel), abilita l'esportazione del filesystem di questo nodo verso agli altri nodi: gli utenti degli altri nodi potranno fare **TUTTO QUELLO CHE VOGLIONO** sul vostro filesystem, a partire da /. Si, potranno fare anche un "rm -rf *" quindi prestate molta attenzione, poiche' di default il filesystem viene esportato: questo perche' si presuppone che sia una sola persona ad amministrare tutti i nodi del cluster e che sappia cio' che sta facendo :-).

Bene, se avete attivato tutti i (o almeno due: io ne ho attivati 4) nodi, passiamo alla prossima fase.

MONITORIAMO IL CLUSTER

Ci sono vari comandi (quelli che abbiamo compilato prima) che aiutano a monitorare, amministrare ed eseguire una fine taratura del sistema. Ecco qui il comando principale:

mosmon

Dopo aver lanciato il comando, appare un grafico che indica il carico (load) dei nodi. I nodi attivi sono indicati dal numero ad essi corrispondente. E' possible avere informazioni premendo dei tasti:

w/v/V/a horizontal (wide), vertical, super-vertical (tight) or automatic selection of numbering
s show processor speeds, rather than loads
m show memory (used out of total) rather than loads
r show memory (raw used/free out of total) rather than loads
u show utilizability percentage rather than loads
l mostra il carico (default)
d mostra i nodi morti (configurati ma non operativi)
D l'opposto del tasto 'd'
t mostra/nasconde il numero dei nodi/CPU (opzione lenta, sconsigliata su cluster grandi)
y mostra lo yardstick in uso (eg. velocita della CPU)
q or Q chiude il programma

Enter ridisegna lo schermo

Se ci sono troppi nodi e non tutti stanno nella schermata:

freccia destra/sinistra muove di un nodo a destra/sinistra
n/p muovono di una pagina a destra/sinistra

TESTIAMO IL CLUSTER

Occorre creare uno script di esempio, eccone uno che lancia 10 processi indipendenti (dei semplici cicli for-next).
(Per chiarezza, rammento che sto sempre lavorando sul nodo master).
Sempre con un text editor, create un file che si chiami test-om.sh e che contenga le prossime cinque righe:

```
#!/bin/bash
#lancia 10 processi awk indipendenti
for n in 0 1 2 3 4 5 6 7 8 9 ; do
awk 'BEGIN { for (i=0;i<10000;i++) for (j=0;j<10000;j++);}' &
done
```

Adesso rendete eseguibile lo script col comando

```
chmod +x test-om.sh
```

Lanciate lo script con:

```
./test-om.sh &
```

Il programma mosmon vi pervetterà di vedere i 10 processi 'sparpagliarsi' per i vari nodi del cluster: inizialmente il nodo che ha lanciato i jobs si carichera' fino a saturare la cpu: dopo qualche decina di secondi, i jobs migreranno sugli altri nodi, per essere eseguiti li'.

ALTRI PROGRAMMI DI MONITORAGGIO

mtop e mps sono le versioni openmosix di top e ps: l'unica visibile differenza e' che mtop e mps mostrano il numero del nodo che sta elaborando il processo.

Viene ora utile il comando:

```
mps a
```

che permette di vedere quanti processi girano e su quali nodi. Notate come nella colonna NODE i numeri varino da 0 a 3, indice del fatto che i processi sono ripartiti tra i 4 nodi del cluster.

```
PID TTY NODE STAT TIME COMMAND

281   2    0 S    0:00 /sbin/getty 38400 tty2
282   3    0 S    0:00 /sbin/getty 38400 tty3
283   4    0 S    0:00 /sbin/getty 38400 tty4
284   5    0 S    0:00 /sbin/getty 38400 tty5
286   6    0 S    0:00 /sbin/getty 38400 tty6
377  ?     0 S    0:00 /bin/cat
378   1    0 S    0:00 -bash
```

```
430  ?     0 S    0:00 /bin/bash
442  ?     0 S    0:00 /bin/bash
453  ?     0 R    0:00 awk BEGIN { for (i=0;i<10000;i++) for (j=0;j<10000;j+
454  ?     1 R    0:00 awk BEGIN { for (i=0;i<10000;i++) for (j=0;j<10000;j+
455  ?     2 R    0:00 awk BEGIN { for (i=0;i<10000;i++) for (j=0;j<10000;j+
456  ?     3 R    0:00 awk BEGIN { for (i=0;i<10000;i++) for (j=0;j<10000;j+
457  ?     3 R    0:00 awk BEGIN { for (i=0;i<10000;i++) for (j=0;j<10000;j+
458  ?     2 R    0:00 awk BEGIN { for (i=0;i<10000;i++) for (j=0;j<10000;j+
459  ?     1 R    0:00 awk BEGIN { for (i=0;i<10000;i++) for (j=0;j<10000;j+
460  ?     1 R    0:00 awk BEGIN { for (i=0;i<10000;i++) for (j=0;j<10000;j+
461  ?     0 R    0:00 awk BEGIN { for (i=0;i<10000;i++) for (j=0;j<10000;j+
462  ?     0 R    0:00 awk BEGIN { for (i=0;i<10000;i++) for (j=0;j<10000;j+
463  ?     1 R    0:00 mps a
```

```
Ecco cosa si vede mandando in esecuzione mtop: anche qui, l'unica
differenza rispetto al comando originale, é l'agginta della colonna 'NODE'.
```

```
PID USER      PRI  NI  SIZE  RSS SHARE STAT N# %CPU %MEM    TIME COMMAND
528 root       20   0   456  456   376 R     3 19.2  0.2    0:05 awk
530 root       20   0   456  456   376 R     2 18.2  0.2    0:05 awk
531 root       19   0   456  456   376 R     2 19.2  0.2    0:05 awk
532 root       19   0   456  456   376 R     3 19.1  0.2    0:05 awk
529 root       20   0   456  456   376 R     1 18.2  0.2    0:05 awk
508 root       14   0   904  904   716 R     0  1.1  0.4    0:03 mtop
```

NOTA: il nodo sul quale ci si trova, viene sempre indicato come nodo zero, indipendentemente da quale sia il suo effettivo numero.

ESEMPIO TIPICO DI CREAZIONE CLUSTER

Supponiamo che abbiate a disposizione una dozzina di PC (aula didattica o ufficio di medie dimensioni) connessi in LAN tramite un HUB (meglio se uno switch): questi PC sono rapidamente (mezza giornata) convertibili in un cluster openMosix a costo zero! Ogni PC può avere il suo kernel con openMosix attivato, e l'utente del PC, tramite la manipolazione del file /etc/openmosix/openmosix.config, è in grado di decidere se:
- abilitare o no il proprio PC come membro del cluster
- richiedere 'potenza elaborativa' al cluster
- fornire 'potenza elaborativa' al cluster
- rendere o no disponibile (esportare) il proprio file-system al cluster (molto comodo per condividere i sorgenti del kernel o altro, ma anche MOLTO pericoloso in quando da remoto possono agire sul filesystem del vostro PC, quindi: **MOLTA ATTENZIONE**)

ALCUNE CONSIDERAZIONI

Come precedentemente indicato, non tutti i programmi traggono beneficio dal clustering, poiché molti sono i fattori che entrano in gioco, come la tipologia di applicativi eseguiti, o la velocità della rete, per esempio.
Nella mia particolare situazione, ho 4 portatili connessi tramite HUB a 10Mbit/s. Questa sistemazione é discreta per determinate situazioni, mentre potrebbe essere deleteria per altre. Supponiamo di dover svolgere questi due compiti:

- CASO_A - calcoli matriciali su 100 vettori (ogni vettore é grande 10KB e genera un risultato di 10KB)
- CASO_B - conversione di 100 file da WAV in MP3 (ogni file WAV é grande 50MB e genera un file MP3 di 5MB)

Per semplificare, supponiamo che:
- il tempo per calcolare un vettore sia uguale al tempo di conversione WAV->MP3 e che questo tempo sia di un minuto
- la banda passante tra un nodo e l'altro sia sempre e comunque di 512KB/s
Ora, se nel CASO_A i tempi di migrazione di un processo da un nodo all'altro é molto piccolo, nel CASO_B il tempo necessario al passaggio del file WAV in andata, sommato al tempo necessario del file MP3 di ritorno, sarebbe superiore al tempo necessario per la trasformazione! Avremmo in questo caso che il cluster ha lavorato più lentamente rispetto al singolo pc, se questo avesse lavorato da solo.
La soluzione più ovvia (ma costosa) é quella di procurarsi hardware più potente (schede di rete a 100Mb o 1000Mb e uno switch o un router per supportare tali velocità di comunicazione, hard disk veloci, molta ram, etc). Inoltre occorre verificare quale sia il collo di bottiblia: non ha senso aumentare la quantita' di ram quando il problema e' imputabile alla lentezza della rete. Avremo modo di verificare dove e come ottimizzare il cluster in un altro documento.

CONCLUSIONI

Siamo arrivati alla fine di questo scritto e in cluster é attivo. Abbiamo visto come oM permetta di 'aggregare' un certo numero di computer per poter ottenere un cluster. Questa molteplicità di macchine viene vista come una unità singola dal punto di vista della potenza elaborativa. Come avevo detto all'inizio, alcuni punti dell'argomento oM sono stati solo accennati, ma sarebbe bene che venissero approfonditi attingendo ai rimandi a cui ho fatto riferimento all'interno di questo documento.

RINGRAZIAMENTI

Un grazie a Bruno Altobelli, per avermi fatto presente alcuni errori/imprecisioni sul documento.

COME E' STATO REALIZZATO QUESTO DUCUMENTO

Ho scritto questo documento usando OpenOffice web writer e poi mcedit per le piccole modifiche. Nessun software closed source e' stato in alcun modo utilizzato per la realizzazione di questo documento.

ASSUNZIONE DI RESPONSABILITA'

Non mi assumo nessuna responsabilita', ne' diretta ne' indiretta, ne' parziale ne' totale, nel caso di guasti parziali o totali e/o di perdita dati parziale o totale degli/dagli elaboratori da voi usati. Chi legge o usa questo documento lo fa a proprio rischio e pericolo. Qualunque danno voi possiate subire dall'utilizzo di questo documento e' di vostra completa responsabilita'.

NOTE SULL'AUTORE

Mi chiamo Roberto Premoli ed ho cominciato ad interessarmi a GNU/Lnux dal 1999.
Sono contattabile agli indirizzi r00(chiocciola)excite.it oppure roberto.premoli(chiocciola)tiscali.it.

Le mie pagine internet sono pubblicate su www.scomodo.com/~roberto e www.newopenbrains.org/roberto