OpenMosix

Cronologia d'installazione di un cluster OpenMosix

(con Slackware 10.2)

di Juri Carlini

e-mail:eth0@slacky.it

Nella stesura di questa mini-guida, considererò che il lettore sappia di cosa si parli ed, eventualmente, abbia dipanato gli ultimi dubbi dando un'approfondita occhiata al mitico e ben scritto "*OpenMosix HOW-TO*", facilmente reperibile in rete anche presso il sito www.openmosix.org, dove viene illustrato con chiarezza cosa sia un "***cluster***" e le diverse tipologie che lo caratterizzano.

Organizzazione generale

Prima di cominciare è bene organizzare il tutto tenendo presente che serviranno almeno due PC (ma più se ne hanno più sarà divertente giocare...) collegabili tra di loro tramite schede *ethernet* che siano quanto meno 100 base "*t*" (100 Mb); non che non sia possibile portare a termine la nostra creazione con delle schede meno veloci ma, lato ludo-didattico a parte, si verrebbero a creare delle restrizioni prestazionali dovute appunto alla bassa velocità alla quale questo tipo di schede lavorano: ci si troverebbe insomma di fronte al cosiddetto "*bottle-neck*" (collo di bottiglia).
Il progetto è stato messo in opera, nel caso specifico, utilizzando :

- 4 PC con CPU AMD (3 Duron ed 1 Athlon)
- 4 schede audio integrate
- 4 schede video "*AGP*" (*Accelerated Graphics Port*)
- Circa 1 Gb di memoria distribuita sui 4 computer
- 4 *hard-disk* (dei quali si può benissimo fare a meno, ma la procedura sarebbe diversa rispetto a quella descritta in questa guida)
- 4 schede *ethernet* "Gigabit" (oramai reperibili ad un costo ragionevole)
- 1 *switch* "Gigabit" *ethernet*

Composizione della "squadra"

Come anche l'autorevole "*OpenMosix HOW-TO*" recita, non è indispensabile avere un elaboratore che si occupi di comandare ed amministrare l'intero *cluster* visto che ognuno dei nodi che lo compongono potrebbe svolgere bene questo compito ma, per praticità, ho deciso di considerare uno di loro come tale (che chiamerò nel corso di tutta la guida "*head*") così da evitare di dover utilizzare un monitor, una tastiera ed un mouse per ognuno di loro tranne nel momento dell'installazione del sistema operativo che ospiterà le varie configurazioni ed i programmi atti a farlo funzionare.
Puramente a titolo informativo, riporto che ho deciso di utilizzare il computer "meno potente" come "head" del mio *cluster* per poterne apprezzare appieno l'efficacia ma ciò, non è particolarmente importante in quanto sarà sempre possibile decidere quale percentuale di carico di lavoro assegnare ad ogni altro elaboratore che lo compone tramite i tools di "*OpenMosix*".
Per praticità, d'ora in avanti, stabiliremo che gli altri PC saranno chiamati sequenzialmente "*nodo1*", "*nodo2*" e "*nodo3*". Ovviamente, potrebbe far piacere a tutti poter avere una connessione con la rete esterna alla nostra (in poche parole Internet) ma, per fare questo, ho deciso di utilizzare un quinto computer che ho chiamato "*gateway*" (un *firewall / gateway* esterno e non facente parte del *cluster*) che ho collegato insieme agli altri tramite lo *switch* ed una scheda di rete con il forwarding dei pacchetti abilitato, ed alla rete esterna tramite un banalissimo *modem* "*56k*". Le premesse sono state fatte... diamo inizio alle danze!

Scelta della distribuzione

Senza ombra di dubbio, dopo innumerevoli prove, la distribuzione che ha servito meglio a questo scopo per semplicità e velocità di configurazione è stata la mitica e massiccia "*Slackware*" nella versione "*10.2*".

ATTENZIONE: la procedura d'installazione del sistema e dei vari pacchetti che vedremo nel corso di questa guida, è da ripetersi per ogni singolo elaboratore che intendiamo far diventare parte del nostro cluster.

PASSO 1: Partizionamento

Partiamo col collegare al primo computer ("head") un monitor, una tastiera, un mouse e, a meno che non si intenda procedere con un'installazione via *NFS* (*Network File System*), un lettore CD-Rom nel quale andremo ad inserire il primo CD d'installazione della distribuzione e dal quale faremo partire la procedura; al *prompt* (se non abbiamo particolari esigenze), premiamo semplicemente [*INVIO*]. Al termine del *boot* del sistema, ci verrà richiesto di selezionare la tastiera adatta alle nostre esigenze (nel nostro caso quella italiana) e poi effettuare il *login*, quindi digitiamo :

slackware login: root **[INVIO]**

Iniziamo ora col partizionare il nostro *hard-disk* (potete farlo come preferite, io ho semplicemente creato una partizione di *swap* da 128Mb all'inizio del disco ed ho assegnato lo spazio rimanente a "/") :

root@slackware:/# fdisk /dev/hd[x] **[INVIO]**

…dove *[x]* cambia chiaramente a seconda del *controller* e della posizione che abbiamo dato al nostro *hard-disk* nel sistema; poi :

Command (m for help): p **[INVIO]**

Se la nostra tabella delle partizioni ci piace così com'è, con un semplice :

Command (m for help): q **[INVIO]**

… usciamo ed andiamo oltre, caso contrario, cancelliamo tutte le partizioni come segue :

Command (m for help): d **[INVIO]**

Command (m for help): 1 **[INVIO]**

Command (m for help): d **[INVIO]**

Command (m for help): 2 **[INVIO]**

… continuando così finché non avremo ripulito completamente la tabella delle partizioni. A cosa fatta, andiamo a crearci le nostre :

Command (m for help): n **[INVIO]**

Partition number (1-4): 1 **[INVIO]**

First cylinder (1-xxx, default 1): **[INVIO]**

Last cylinder or +size or +sizeM (1-xxx, default xxx): +128M **[INVIO]**

Command (m for help): t **[INVIO]**

Partition number (1-4): 1 **[INVIO]**

Hex code (type L to list codes) : 82 **[INVIO]**

Command (m for help) : n **[INVIO]**

Partition number (1-4): 2 **[INVIO]**

First cylinder (xx-xxx, default xx): **[INVIO]**

Last cylinder or +size or +sizeM (xx-xxx, default xxx): **[INVIO]**

Command (m for help): a **[INVIO]**

Partition number (1-4): 2 **[INVIO]**

Command (m for help): p **[INVIO]**

Controlliamo che tutto sia OK, poi :

Command (m for help): w **[INVIO]**

Una volta fatto questo, se abbiamo cambiato la tabella delle partizioni, è sempre buona norma fare un *reboot* del sistema; personalmente non ho mai incontrato problemi, ma con dell'hardware un po' più datato del mio, ho notato che ciò è fondamentale. Possiamo quindi riavviare il computer con la pressione di [*CTRL*] + [*ALT*] + [*CANC*].

PASSO 2: Installazione

Dopo l'eventuale e consigliato *reboot*, ci verrà ripresentato di nuovo il solito *prompt* :

slackware login: root **[INVIO]**

Ci troveremo di nuovo a dover impostare la nostra tastiera dopodiché, al successivo *prompt*, digiteremo :

root@slackware:/# setup **[INVIO]**

... ed andiamo al passo dell'installazione che ci permette di aggiungere la partizione di *swap* che abbiamo creato in precedenza scegliendo, dal menu che ci si presenta, la voce "*ADDSWAP*". Selezioniamo ora la partizione adatta (che normalmente ci viene già proposta) e la formattiamo per far si che poi il sistema automaticamente la attivi e la monti.
Di seguito, tramite la tab "*TARGET*", andiamo a selezionare la partizione nella quale vogliamo installare il sistema per poi procedere con il verificare se quella proposta è quella giusta, scegliere il tipo di *filesystem* da utilizzare (io personalmente non abbandono mai "*reiserfs*") ed infine formattarla. Il passo successivo, è quello di determinare il supporto dal quale ricavare l'installazione per cui, nel nostro caso, avendo un CD-Rom a disposizione, faremo sì che il programma d'installazione imposti automaticamente quest'ultimo come sorgente.
Adesso dobbiamo decidere cosa installare; di default, ci troveremo selezionati tutti i pacchetti di maggior importanza che andranno controllati ed aggiunti e/o tolti a seconda delle nostre necessità. Per quanto mi riguarda, trovo sia particolarmente importante installare anche il pacchetto "*KDEI*" che ci permetterà, in futuro, di poter scegliere la lingua italiana come lingua predefinita nel *desktop environment* "*KDE*" (*K Desktop Environment*), se intendiamo utilizzare quest'ultimo *desktop manager*.
Fatto questo, si deve decidere che tipologia d'installazione adottare; personalmente, scelgo sempre la possibilità di selezionare i pacchetti manualmente tramite la voce "*menu*" in quanto dà modo di poter avere un quadro abbastanza chiaro dei vari pacchetti che ci vengono proposti e fornisce anche una breve descrizione di questi ultimi senza per questo essere un metodo troppo dispendioso in termini di tempo.
Un attimo di attenzione: non è scopo di questa guida fornire una spiegazione passo passo sull'installazione di un sistema *GNU/Linux*, pertanto sarebbe opportuno porre la massima attenzione a ciò che si sceglie di inserire od appoggiarsi a delle guide di riferimento, in quanto ci si potrebbe trovare con qualche pacchetto fondamentale in meno e, di conseguenza, non riuscire a svolgere tutti i passaggi di seguito illustrati che sono vitali per poter, alla fine, godere di un *cluster* nuovo fiammante. Consiglierei a chi non si sentisse a suo agio con un'installazione a "*menu*" di *Slackware*,

di rimandare la decisione di beneficiare di un *cluster* a giorni futuri, ma per coloro che si sentono particolarmente coraggiosi, posso dare delle linee guida da far rabbrividire i puristi; di seguito alcuni dettagli da includere assolutamente :

- Tutti i compilatori ("*gcc*")
- Tutte le librerie di sistema (anche quelle che non si conoscono)
- Il sistema grafico "*X*" (almeno per il computer "head")
- Tutte le cose che possano permettere un funzionamento impeccabile del protocollo *TCP/IP*
- Il server "*sshd*"
- KDE

Di seguito invece, alcune cose delle quali possiamo benissimo fare a meno nell'implementazione del nostro *cluster* o semplicemente servizi che possiamo non attivare :

- Il sistema di stampa se consideriamo di non dover utilizzare una stampante
- Il servente *DNS*
- Il server *DHCP* (visto che gli indirizzi verranno assegnati in maniera statica)
- Tutti i vari server di posta se intendiamo utilizzare quelli offerti da Internet

A questo punto, dopo un'attesa più o meno lunga, giungeremo alla fine dell'installazione dei pacchetti e la finestra seguente ci chiederà quale *kernel* vogliamo installare per poter far funzionare il sistema; noi, sia per semplicità, sia perché ci ha servito bene durante la procedura d'installazione, opteremo per quello che si trova nel CD d'installzione (il primo, nel caso in cui durante la procedura si sia dovuto inserire il secondo) e selezioneremo la voce "*cdrom*" indicandogli di installare il *kernel* chiamato "*bare.i*".
Di seguito, ci verrà chiesto se intendiamo creare un floppy di *boot* (cosa sempre buona da fare), rispondiamo ed andiamo avanti.
Il prossimo passaggio, sarà la configurazione del *modem*; noi, non ne abbiamo (come detto all'inizio, sarebbe il caso di non accedere ad Internet direttamente per mezzo del *cluster* ma utilizzare un PC come *gateway*), per cui selezioneremo la voce "*no modem*".
Andando oltre, ci vedremo chiedere se intendiamo attivare il sistema di "*hotplug*": essendo di vitale importanza, decidiamo di farlo rispondendo "*yes*" e proseguiamo (se si preferisce, si potrà sempre disabilitarlo in seguito, una volta sicuri che tutto funzioni, aggiungendo manualmente i moduli che ci servono agli *script* di avvio).
La successiva schermata ci propone l'installazione di "*LILO*" (*LInux LOader*) e, per ora, selezioniamo la modalità d'installazione "*simple*" che va benissimo anche in considerazione del fatto che, in seguito, dovremo andare a modificarlo secondo le nostre esigenze.
Facendo un ulteriore passo in avanti, ci verrà chiesto quale risoluzione video adottare durante la visualizzazione del *bootstrap* del sistema (se vogliamo cioè attivare il "*frame buffer*" o meno); direi che la questione in oggetto non ci interessa particolarmente giacchè, nell'utilizzo finale, non vedremo a schermo il *boot* di 3 computer su 4, per cui, selezioniamo semplicemente la voce "*standard*" ed andiamo oltre (anche quest'ultima impostazione, potrà essere cambiata in seguito tramite un apposito parametro da passare a *LILO*).
A questo punto, saremo interpellati per quanto riguarda la configurazione minimale di *LILO* e dovremo fargli sapere se il CD-Rom dal quale stiamo installando il sistema ci servirà in futuro anche come masterizzatore (o, se non questo, quale intendiamo utilizzare se ne abbiamo uno) o se, per esempio, il nostro bus di sistema è superiore a 33Mhz, per cui dovremo inserire i parametri adatti a tale scopo (es. "*hdc=ide-scsi*" se mettiamo il masterizzatore come "*master*" nel secondo *controller* "*IDE*" (*Integrated Drive Electronics*) ecc. ecc. e "*idebus=66*" se supportato dalla nostra mainboard). Di seguito, il programma d'installazione, ci chiederà dove vogliamo installare *LILO*; personalmente, trovo che la cosa più sensata sia di installarlo nell'"*MBR*" (*Master Boot Record*) in quanto sarà dall'*hard-disk* che il nostro sistema si dovrà avviare.
Andiamo poi a configurare il mouse e selezioniamo "*ps/2*" se abbiamo un semplice mouse senza rotellina, "*imps/2*" se invece disponiamo di un mouse con la rotellina e via dicendo.
Il prossimo passo ci porterà a scegliere se desideriamo o meno avere la possibilità di effettuare il famoso "*copia/incolla*" anche dal terminale: siccome può tornare spesso comodo, decidiamo di rispondere "*yes*" ed attivare questa funzione.

Ora passeremo a configurare i servizi di base della rete; in questa fase, dovremo andare a configurare la nostra rete tenendo sempre a mente che tutto ciò che impostiamo in questa fase, potrà sempre essere cambiato in futuro con semplicità ma che, se lo facciamo bene con un minimo d'attenzione da ora, dovremo lavorare un pochino meno in seguito... per cui, rispondiamo "*yes*" alla configurazione della rete e successivamente inseriamo il nome che abbiamo deciso di dare all'elaboratore che stiamo installando (abbiamo detto "head" e gli altri di seguito "nodo1", "nodo2" ...) ed andiamo avanti.
A questo punto si dovrà scegliere un nome di domino (utilizzeremo come esempio "*cluster*") al quale far appartenere tutti i computer della nostra rete (a meno che per esigenze particolari non si debbano creare più sottoreti) e che dovrà essere uguale per *tutti* i nodi, compreso l'eventuale PC che ci dovrà fornire l'accesso ad Internet.
Nella schermata successiva dovremo selezionare la voce "*static IP*" in quanto, per semplicità e sicurezza, abbiamo scelto in precedenza, di assegnare un indirizzo di rete statico ai nostri nodi ed inseriremo quindi un indirizzo valido per una rete privata (es. 192.168.1.100 per "head", 192.168.1.101 per "nodo1", 192.168.1.102 per il "nodo2" e via dicendo) ed andiamo avanti.
Adesso, ci verrà chiesto l'indirizzo del *gateway* che ci darà accesso ad Internet per cui, se lo abbiamo già predisposto, inseriremo quello, altrimenti ne sceglieremo uno che non intendiamo utilizzare per gli altri nodi del *cluster* ma che sarà l'indirizzo di rete esclusivo del computer "gateway".
Passiamo ora alla fase successiva; si tratta di attivare i servizi di rete che ci potranno interessare. Tra tutti, a mio riguardo, ho ritenuto utili :

- "*inetd*"
- "*smbd*"
- "*nfsd*"
- "*sshd*"

…e, nel caso si stia installando il computer "gateway", anche :

- "*ip_ forward*".

A questo punto la configurazione di rete è completata e ciò ci agevolerà non di poco il lavoro futuro.
Nella prossima schermata, dovremo confermare di non voler testare nessun "*font*" di caratteri, selezionando "*no*" e procedendo oltre; ancora di seguito dovremo dire al programma d'installazione che non ci troviamo in corrispondenza del "*GMT*" (*Greenwich Mean Time*) e scegliamo la zona geografica adatta ("*Europe/Rome*") per arrivare infine a decidere quale *desktop environment* adottare come default; nel mio caso ho scelto *KDE*.
E' quindi giunto il momento di scegliere una *password* per l'utente *root* e ci verrà chiesto di digitarla due volte per poter poi proseguire ed arrivare fino alla... "fine dell'inizio". La strada da percorrere è ancora molto lunga, ma ora, ci troviamo con un sistema completo che dovrebbe essere in grado di riavviarsi con la pressione dei tasti [*CTRL*] + [*ALT*] + [*CANC*].
Scontato ripetere che la procedura appena descritta non è altro che la semplice (o meno) installazione di *Slackware* e che *va ripetuta per tutti i computer* che intendiamo far diventare nodi del nostro *cluster*; modifichiamo a nostro piacimento soltanto la parte relativa alla configurazione di rete assegnando un nome ed un indirizzo *IP* (*Internet Protocol*) differente ad ogni elaboratore e facendo attenzione ad inserire lo stesso nome di dominio e lo stesso indirizzo del computer che farà da *gateway* a tutti.
Andiamo per cui ad effettuare il primo *reboot* della macchina.

PASSO 3: Configurazione dei servizi di bootstrap

Una volta che il sistema si è riavviato e ci ha ripresentato il *login*, dobbiamo andare ad editare i vari file di configurazione e modificarli per poter snellire i futuri processi di *bootstrap*. Personalmente, ritengo fondamentale poter lavorare con un editor di testi basilare, leggero e veloce, quale ad esempio "*nano*" che però, purtroppo, non si trova nei CD d'installazione. Se anche voi la pensate

come me, scaricate il pacchetto in formato "*.tgz*" da http://www.slacky.eu/repository/slackware-11.0/office/nano/2.0.4/ ed andiamo ad installarlo con :

root@head:~# installpkg nano-2.0.4.-i486-1sl.tgz **[INVIO]**

A questo punto siamo pronti; gli script che ci interessa modificare si trovano nella directory "*/etc/rc.d/*" :

root@head:~# cd /etc/rc.d/ **[INVIO]**

Diamo un'occhiata ai vari file che ci sono :

root@head:/etc/rc.d/# ls **[INVIO]**

Andiamo ora ad editare e commentare tutte le linee di codice che permettono l'attivazione di quei servizi che non ci interessa far partire durante il *boot* del sistema (ad esempio, inutile dirgli di far partire il database "*mysql*" se non lo abbiamo nemmeno installato così come "*Apache*" ecc. ecc.) :

root@head:/etc/rc.d/# nano rc.S **[INVIO]**

root@head:/etc/rc.d/# nano rc.6 **[INVIO]**

root@head:/etc/rc.d/# nano rc.K **[INVIO]**

root@head:/etc/rc.d/# nano rc.M **[INVIO]**

root@head:/etc/rc.d/# nano rc.inetd2 **[INVIO]**

root@head:/etc/rc.d/# nano rc.inet1 **[INVIO]**

root@head:/etc/rc.d/# nano rc.inetd.conf **[INVIO]**

Ovviamente non c'è una regola precisa nella modifica degli *script* visti sopra ed esula anche dall'intenzione di questa guida, ma c'è una considerazione da fare: a parte il nostro "head", sul quale verranno gestite la configurazione e le prestazioni dell'intero *cluster* e al quale collegheremo i nostri unici monitor, tastiera eccetera, degli altri nodi, una volta configurati a dovere uno per uno, non vedremo nemmeno i messaggi di *bootstrap* per cui, più quest'ultimo sarà snello e veloce, più sarà pratico e semplice l'utilizzo del *cluster*.
Assicuriamoci di non aver commesso errori controllando che il sistema si riavvii senza intoppi; nel caso ce ne fossero, aiutiamoci leggendo con attenzione i messaggi che la nostra *Slackware* ci stampa a schermo e/o aiutandoci con il suo dettagliatissimo manuale.

PASSO 4: Patching del kernel

Una volta appurato che tutto sia OK, possiamo cominciare con il cercare la versione del *kernel* che fa al caso nostro ed i pacchetti necessari. Il *kernel* che ci serve (l'ultima versione disponibile al momento in cui sto scrivendo) è il "*vanilla-kernel*" scaricabile dal sito www.kernel.org nominato :

" linux-2.4.26.tar.gz " o **" linux-2.4.26.tar.bz2 "**

I pacchetti invece che ci interessano, per poter far sì che il nostro sistema diventi un *cluster* "*OpenMosix*", sono :

" openmosix-2.4.26-1.bz2 "

...e :

“ openmosix-tools-0.3.6-2.tar.gz ”

...con l’aggiunta di :

“ openmosixview-1.5.tar.gz ”

Andiamo ora a dare una breve spiegazione di ognuno di loro.
La versione del *kernel* che ci siamo scaricati è quella alla quale si deve applicare l’ultima “*patch*” testata e stabile chiamata “*openmosix-2.4.26.bz2*” che permette, previa riconfigurazione del *kernel* stesso, di abilitare tutte quelle voci che rendono possibile avere la piena funzionalità che un *cluster* “*OpenMosix*” offre.
Il secondo pacchetto invece, come anche il nome stesso lascia intuire, “*openmosix-tools*”, serve a fornire tutti quegli strumenti coi quali possiamo amministrare e configurare al meglio, da linea di comando, il nostro *cluster*. E’ grazie a questi infatti, che ci potremo sbizzarrire a far “girare” un determinato processo su di un elaboratore anziché su un altro o, semplicemente, monitorare cosa stia facendo il nostro *super-computer* in un determinato momento. L’ultimo pacchetto che ci siamo procurati, invece, altro non è che l’interfaccia grafica degli strumenti di amministrazione visti in precedenza grazie al quale ci sarà possibile gestire il *cluster* anche dai nostri ambienti grafici preferiti (nella fattispecie, “*openmosixview*” si appoggia alle librerie “*QT*”, le native di *KDE*) e considerare il tutto come un gioco entusiasmante. Ma passiamo alla pratica.
Portiamoci nella directory che dovrà contenere i sorgenti del *kernel* con un :

root@head:~# cd /usr/src/ **[INVIO]**

Qui, se durante la fase di installazione del sistema abbiamo deciso di metterci anche i sorgenti del *kernel* “*bare.i*”, troveremo una directory (oltre ad eventuali altre) chiamata “*linux-2.4.31*” ed un *link* chiamato “*linux*” che punta alla directory in questione; controlliamo ciò con un :

root@head:/usr/src/# ls -all **[INVIO]**

Una volta individuato il *link* “*linux*”, rimuoviamolo, in quanto avremo bisogno di crearne uno nuovo che dovrà puntare alla directory che si creerà quando andremo a decomprimere l’archivio coi sorgenti del *kernel* che abbiamo scaricato, quindi :

root@head:/usr/src/# rm linux **[INVIO]**

Ora, supponendo che ci siamo ritrovati coi pacchetti depositati nella directory “*/root/*”, occorrerà eseguire :

root@head:/usr/src/# tar -xvzf /root/linux-2.4.26.tar.gz **[INVIO]**

...oppure :

root@head:/usr/src/# tar -xvjf /root/linux-2.4.26.tar.bz2 **[INVIO]**

Una volta fatto ciò, bisognerà provvedere a ricreare il *link* che avevamo tolto in precedenza e lo faremo puntare alla nuova directory che si è creata :

root@head:/usr/src/# ln -s /usr/src/linux-2.4.26 linux **[INVIO]**

A questo punto, siamo pronti a fare il passo che ci avvicinerà in maniera irreversibile verso la messa in opera del nostro *cluster*; dobbiamo “patchare” i sorgenti che abbiamo scompattato e “linkato” con il file “*openmosix-2.4.26-1.bz2*”. Per farlo, procederemo copiando nella directory contenente i sorgenti del *kernel* in questione, il file di cui sopra :

root@head:/usr/src/# cp /root/openmosix-2.4.26-1.bz2 /usr/src/linux/ **[INVIO]**

Dopodichè, per applicare la *patch*, dovremo entrare nella directory del nostro nuovo *kernel* digitando :

root@head:/usr/src/# cd linux **[INVIO]**

...e di seguito, opereremo in questo modo :

root@head:/usr/src/linux/# bzcat openmosix-2.4.26-1.bz2 | patch -Np1 **[INVIO]**

Il gioco è fatto; ora, non dobbiamo fare altro che andarci a ricompilare a dovere il nostro bel *kernel*; la *patch* che abbiamo appena applicato, sarà presente nel menu della riconfigurazione come prima voce della lista (chiamata, ovviamente, "*OpenMosix*").
Procediamo con la normale ricompilazione con :

root@head:/usr/src/linux/# make mrproper **[INVIO]**

…poi :

root@head:/usr/src/linux/# make menuconfig **[INVIO]**

Successivamente, entriamo all'interno della voce che ci interessa ed abilitiamo le seguenti opzioni :

[*] openMosix process migration support

[ ] Support clusters with a complex network technology

[ ] Stricter security on openMosix ports

(3) Level of process-identity disclosure (0-3)

[*] Poll/Select exceptions on pipes

[ ] Disable OOM killer

[*] Enable Extension: Local Time

FATE ATTENZIONE! Le opzioni attivate sotto la voce "openMosix" devono tassativamente essere le stesse per tutti i nodi che compongono il cluster !!!

Ovviamente, quelle riportate sopra, sono le opzioni che ho attivato io, in base alle mie esigenze; purtroppo, anche per quanto riguarda la scelta delle opzioni per "*OpenMosix*", vale la stessa regola che vale in generale quando si va a ricompilare un *kernel*, ossia la conoscenza del proprio hardware, la propria rete, la consapevolezza delle proprie esigenze eccetera, per cui è impossibile stabilire una linea guida generale che possa andare bene a tutti indistintamente.
Una volta esserci assicurati di aver eseguito tutto a dovere usciremo dal "*menuconfig*" rispondendo "*yes*" alla richiesta di salvataggio della configurazione e procederemo come da manuale con :

root@head:/usr/src/linux/# make dep **[INVIO]**

…poi :

root@head:/usr/src/linux/# make clean **[INVIO]**

…quindi :

root@head:/usr/src/linux/# make bzImage **[INVIO]**

Di seguito creeremo i moduli (se abbiamo scelto di abilitarli) :

root@head:/usr/src/linux/# make modules [INVIO]

…ed andremo ad installarli con :

root@head:/usr/src/linux/# make modules_install [INVIO]

A questo punto, copiamo l'immagine creatasi nella directory di *boot* ("/boot/") :

root@head:/usr/src/linux/# cp arch/i386/boot/bzImage /boot/cluster [INVIO]

…e successivamente, controlliamo che nella directory "*/boot/*" non ci sia già un file od un *link* chiamato "*system.map*" :

root@head:/usr/src/linux/# ls -all /boot/ [INVIO]

Nel caso fosse un semplice *link*, possiamo rimuoverlo con :

root@head:/usr/src/linux/# rm /boot/system.map [INVIO]

Se al contrario si trattasse di un file vero e proprio, sarebbe bene non liberarsene definitivamente ma rinominarlo in qualcosa di differente (diciamo "*old_system.map*" ad esempio) :

root@head:/usr/src/linux/# mv /boot/system.map /boot/old_system.map [INVIO]

A questo punto siamo pronti a copiarci il nostro :

root@head:/usr/src/linux/# cp system.map /boot/ [INVIO]

Fatto ciò, andiamo ad editare il file di configurazione d'avvio per aggiungerci la possibilità di avviare il *kernel* "*OpenMosix*" che abbiamo appena creato :

root@head:/usr/src/linux/# nano /etc/lilo.conf [INVIO]

…e alla fine, ci aggiungeremo le seguenti linee :

…

image = /boot/cluster
root = /dev/hdX
append = "hdX=ide-scsi idebus=66"
label = kernel_mosix
read-only

Usciamo dall'editor controllando di non aver fatto alcun danno con :

root@head:/usr/src/linux/# lilo -t [INVIO]

...e se l'output del precedente comando non ha dato errori, procediamo con :

root@head:/usr/src/linux/# lilo -v [INVIO]

...seguìto da un incrocio di mani a mo' di preghierina e da un bel :

root@head:/usr/src/linux/# reboot [INVIO]

Il computer, con immensa paura da parte nostra, si riavvierà, noi selezioneremo al *prompt* di *LILO* il nostro *kernel* che abbiamo chiamato "*kernel_mosix*" nuovo fiammante e, visto che la compilazione è stata perfetta (... ! ...), ci ritroveremo al *prompt* di *login* in me che non si dica!

PASSO 5: Installazione di "OpenMosix"

Effettuiamo il *login* come *root* ed andiamo ad installare ora i tools precedentemente scaricati che ci permetteranno di amministrare il *cluster*. Faremo :

root@head:~# cd /usr/src/ [INVIO]

...ed una volta lì :

root@head:/usr/src/# tar -xvzf openmosix-tools-0.3.6-2.tar.gz [INVIO]

…dopodichè, come da prassi, per l'installazione di un semplice programma procederemo con lo spostarci all'interno della directory che il precedente comando ha creato con :

root@head:/usr/src/# cd openmosix-tools [INVIO]

…ed impartiremo :

root@head:/usr/src/openmosix-tools/# ./configure --with-kerneldir=/usr/src/linux-2.4.26

[INVIO]

...creeremo l'eseguibile :

root@head:/usr/src/openmosix-tools/# make [INVIO]

…e lo installeremo con :

root@head:/usr/src/openmosix-tools/# make install [INVIO]

...e, visto che non fa mai male aggiornare le librerie di sistema dopo aver installato qualche pacchetto, lanceremo :

root@head:/usr/src/openmosix-tools/# ldconfig [INVIO]

PASSO 6: Configurazione di "OpenMosix"

Se tutto è andato bene fin qui, dobbiamo fare gli ultimi ritocchi ai vari file di configurazione.
Tenendo conto che la procedura d'installazione del sistema, del *kernel* "O*penMosix*" e dei tools di amministrazione fin qui visti è la *medesima per ogni nodo del cluster*, dobbiamo andare a modificare il file "*/etc/hosts*" in modo che risulti *uguale in tutti i nodi*; per farlo, procederemo così :

root@head:/usr/src/openmosix-tools/# nano /etc/hosts [INVIO]

…ed il file (sempre considerando che durante l'installazione dei sistemi abbiate tenuto conto di questa guida) dovrà contenere le seguenti linee :

…

127.0.0.1 localhost
192.168.1.100 head.cluster head

192.168.1.101 nodo1.cluster nodo1
192.168.1.102 nodo2.cluster nodo2
192.168.1.103 nodo3.cluster nodo3

…

Ora, per avere una visione d'insieme, portiamoci nella directory "*/etc/*" :

root@head:/usr/src/openmosix-tools/# cd /etc/ **[INVIO]**

…e quindi :

root@head:/etc/# ls **[INVIO]**

Come possiamo vedere, troveremo nella directory un file chiamato "*openmosix.map*" ed, editandolo, andremo ad aggiungervi alla fine della parte commentata, le seguenti righe di configurazione :

…

1 head 1
2 nodo1 1
3 nodo2 1
4 nodo3 1

…

Chiudiamo l'editor salvando il file e *ricordiamoci che questo file, come anche il precedente, dovrà essere* ***identico*** *per tutti i nodi*.
A questo punto, andiamo ad editare anche l'ultimo file di configurazione per fare in modo che automaticamente, all'avvio, ogni singolo elaboratore entri a far parte del nostro *cluster*; per fare questo, aggiungeremo il comando corrispondente allo script "*/etc/rc.d/rc.local*" :

root@head:/etc/# nano rc.d/rc.local **[INVIO]**

...ed inseriamoci il comando :

…

setpe -w -f /etc/openmosix

…

OK, a questo punto possiamo riavviare tutti i computer e, se tutto è stato fatto a dovere, il nostro *cluster* sarà già funzionante appieno ed i processi liberi di migrare da un nodo all'altro.

PASSO 7: Testiamo il cluster

Grazie agli "*openmosix-tools*" precedentemente installati, possiamo già cominciare a giocare con la nostra nuova fiammante "cluster-mega-linux-box"; ad esempio, lanciando in locale il comando "*mosmon*" :

root@head:~/# mosmon **[INVIO]**

...potremo visualizzare una schermata che ci tiene aggiornati sul carico del nostro *cluster*, permettendoci di visualizzare il dettaglio di ogni nodo appartenente ad esso sull'ascissa del grafico ed il relativo carico sull'ordinata; altro simpatico strumento di gioco è il comando che permette di

far migrare una determinata applicazione su di un nodo differente rispetto a quello in cui sta girando in questo momento, ad esempio :

root@head:~/# migrate *[xxx]* nodo3 **[INVIO]**

...dove "*[xxx]*" sta per il "*pid*" (*Process Identification Number*)del processo che si vuole far migrare. Da notare che non tutti i processi, ovviamente, hanno la possibilità di migrare. Ad esempio, l'applicativo "*Grip*" che permette di codificare i CD audio in "*MP3*", non potrà migrare in quanto ha bisogno di interagire con il lettore CD/DVD per poter estrarre le tracce. Al contrario, migrerà "*LAME*" che è appunto l'applicazione che effettua la codifica da traccia audio ad "*MP3*"!.
Felici ed entusiasti di aver ottenuto ciò, c'è, però, un appunto da fare; tutto questo è già molto bello di per sé, ma le impostazioni di default non è detto che conoscano ed abbiano settato al meglio tutti i nodi (anche se probabilmente sarà così). Nel caso in cui, ad esempio, avessimo un nodo molto più potente rispetto al nostro "head" (sul quale, mettiamo, si stia lavorando) quanto a quantitativo di memoria e velocità della *CPU*, potremmo desiderare che la maggior parte dei processi che andremo a lanciare, migrino sul nodo in questione almeno fino al suo utilizzo massimo. Per fare questo, ci affideremo ancora una volta ai tools "*OpenMosix*" :

root@head:~/# mosctl getspeed **[INVIO]**

…che, se lanciato in locale, ci darà come output un valore numerico rappresentante la "velocità" che "*OpenMosix*" ha stabilito per l'elaboratore sul quale stiamo lavorando.
N.b.: questa "velocità" non è una vera e propria velocità né di rete, né di elaborazione, ma semplicemente un valore che permette di impostare la priorità di un nodo rispetto agli altri.
Se non ci piace, possiamo variarla settando un valore più alto o più basso, dopo aver analizzato anche gli altri, con il comando :

root@head:~/# mosctl setspeed *[n]* **[INVIO]**

…dove *[n]*, sarà appunto il valore in questione che sceglieremo.

PASSO 8: Configurazione mediante interfaccia grafica

Allora, se tutto ciò ci ha reso felici a sufficienza, interrompiamo il momento di svago e passiamo adesso a configurare un sistema di gestione dei settaggi del nostro *cluster* con il supporto dell'interfaccia grafica. Non che non sia abbastanza efficiente tutto quello che abbiamo visto fin qui, ma poterlo fare semplicemente spostando delle levette (proprio come se fosse un mixer audio) oppure cliccando su un bottone, è certamente più comodo, pratico e soprattutto bello.
Per fare questo, dovremo ritornare dove abbiamo precedentemente deciso di scompattare i sorgenti che installiamo; per cui procederemo così :

root@head:~/# cd /usr/src/ **[INVIO]**

...e lanceremo di nuovo :

root@head:/usr/src/# tar -xvzf /root/openmosixview-1.5.tar.gz **[INVIO]**

Al solito, il precedente comando avrà creato la directory corrispondente al pacchetto da installare e noi dovremo portarci al suo interno per poter configurare i sorgenti nel solito seguente modo :

root@head:/usr/src/# cd openmosixview-1.5 **[INVIO]**

…e di seguito :

root@head:/usr/src/openmosixview-1.5/# ./configure **[INVIO]**

Dovrebbe essere andato tutto per il meglio, per cui insistiamo con un :

root@head:/usr/src/openmosixview-1.5/# make [INVIO]

...ed ancora :

root@head:/usr/src/openmosixview-1.5/# make install [INVIO]

Per rifinire e concludere il tutto :

root@head:/usr/src/openmosixview-1.5/# ldconfig [INVIO]

A questo punto, se tutto è andato bene (e mi auguro che lo sia), dovremmo poter lanciare il programma con un semplicissimo comando :

root@head:/usr/src/openmosixview-1.5/# openmosixview & [INVIO]

Impressionante vero? Purtroppo però (c'è sempre un però!), ci accorgeremo ben presto che, se tentassimo di impostare la famosa "velocità" vista in precedenza in maniera da settare la priorità di un nodo rispetto ad un altro a nostro piacimento, ci apparirà, in basso a sinistra nella finestra, una dicitura del tipo "*cannot set speed on 192.168.1.101*" e la levetta di cui sopra, tornerà impietosamente indietro... questo non è buono, per cui decidiamo di chiudere il programma.
Il problema che si viene a creare è dovuto al fatto che quest'ultimo, di default, ha bisogno di sapere "come" deve comunicare con gli altri nodi. Per ragioni di sicurezza e per praticità (visto che in rete c'è abbondante documentazione a riguardo) decidiamo che lo scambio di informazioni con gli altri computer, debba avvenire tramite il famosissimo "*SSH*".
Per fare questo, la procedura più semplice (non l'unica, non la migliore) è di impartire il seguente comando :

root@head:/usr/src/openmosix-1.5/: ssh-keygen -t rsa [INVIO]

Il programma "*ssh-keygen*" ci chiederà una passphrase; digitiamola correttamente e ripetiamola se preferiamo, altrimenti facciamo semplicemente due volte [*INVIO*] senza inserirne alcuna. Quest'ultimo comando ci permette di creare una "chiave" (una sorta diciamo di "file di riconoscimento") che, una volta salvata in locale e copiata in remoto nei vari nodi, garantisce che l'identità di chi sta cercando di comunicare da un nodo all'altro, sia la nostra, per cui non ci verrà richiesta nessuna *password* per l'autenticazione (anche in considerazione del fatto che tramite il programma non potremmo farnirla).
Per far ciò, procederemo con :

root@head:~/# scp /root/.ssh/id_rsa.pub nodo1:/root/.ssh/authorized_keys2 [INVIO]

…e ripeteremo il commando dato sopra per ogni singolo nodo. Andremo quindi a farlo anche per l'"head" stesso con :

root@head:~/# cp /root/.ssh/id_rsa.pub /root/.ssh/authorized_keys2 [INVIO]

…ed il gioco è fatto! Controlliamo che tutto vada come dovrebbe tentando di effettuare un *login* remoto o semplicemente rilanciando :

root@head:~/# openmosixview & [INVIO]

Ora, dovremmo poter settare le differenti velocità di ogni nodo (ivi compreso il nostro) e tante altre cose tramite interfaccia grafica.
Preciso a questo punto che l'installazione del pacchetto "*openmosixview*", non sarà necessaria su quei nodi che intendiamo utilizzare senza il server grafico "*X*".

PASSO 9: Considerazioni finali

Le considerazioni che potremmo fare sulle ottimizzazioni da apportare al nostro *cluster* "*OpenMosix*" sono molteplici: a partire dagli applicativi da utilizzare per sfruttarlo al meglio, fino ad arrivare alle varie configurazioni di rete che potremmo implementare; leggevo proprio l'altro giorno della documentazione che prendeva in esame proprio quest'ultimo punto. A quanto pare (ed è anche facilmente immaginabile) un *cluster* che abbia i vari nodi connessi in modalità "*P2P*" (*peer-to-peer*), rende molto di più in termini di migrazione dei processi (e quindi di resa generale delle elaborazioni) rispetto ad un *cluster* configurato tramite *switch* soprattutto se la rete in oggetto è a 100Mb. È chiaro che la configurazione "*P2P*" comporta dei limiti quali, ad esempio, il numero massimo di dispositivi *ethernet* configurabili in ciascuno dei nodi che dipende direttamente dal numero di slots *PCI* disponibili sulle mainboards (sebbene esistano in commercio delle meravigliose schede ethernet con 4 o più porte anche "*Gigabit*").
Se fate esperimenti in proposito fatemi sapere, sarebbe interessante scambiarci informazioni ed esperienze a riguardo!

Buon divertimento !

Per suggerimenti, correzioni e/o aggiunte, Vi prego di inviare un'e-mail a : eth0@slacky.it .