

Cluster ad alta disponibilità basato su Gentoo

Enrico Morelli

13 Luglio 2006

Abstract

Questa guida mostra come installare un cluster ad alta disponibilità basato su Gentoo e composto da due computer.

Indice

1	Introduzione	7
2	Architettura hardware, versioni del software e accessori	9
3	Installazione di Gentoo	11
3.1	Caricamento dei moduli	11
3.2	Setup delle partizioni	11
3.3	Il file raidtab	12
3.4	Creazione dei filesystem	13
3.5	Mount delle partizioni	14
3.6	Note finali: kernel, fstab, grub, pacchetti aggiuntivi e rete	15
4	Replicare l'installazione	17
5	DRBD: preliminari	19
5.1	Installazione del channel-bonding	19
5.1.1	Introduzione	19
5.1.2	Installazione	19
5.1.3	Installazione di pconsole	21
5.2	Installazione del DRBD	22
5.3	Configurazione del DRBD	22
6	Heartbeat	25
6.1	Preliminari	25
6.1.1	Resource Group	25
6.2	Installazione	26
6.3	Configurazione	26
6.3.1	Il file haresources	26
6.3.2	Il file ha.cf	26
6.3.3	Il file authkeys	28
6.3.4	Partenza di Hearbeat	28
6.3.5	Test	28

7	Troubleshooting	31
7.1	RAID e DRBD	31
7.2	Heartbeat e DRBD	31

Capitolo 1

Introduzione

I requisiti per questa guida sono, chiaramente, la conoscenza di Gentoo, dei concetti che sono dietro al concetto di RAID e un paio di computer.

Brevemente, il RAID 1, o **mirroring**, consta di almeno due dischi che vengono combinati in un volume della dimensione del disco più piccolo. I vantaggi sono dati dal fatto che se si rompe uno dei due dischi non si perdono i dati ed il sistema continua a funzionare.

DRBD¹ è l'acronimo di **Distributed Replicated Block Device** ed è sviluppato da Philipp Resiner. Drbd è un dispositivo a blocchi creato per risolvere problemi tipici inerenti cluster in alta disponibilità. Si basa sul meccanismo della replicazione dei dati fra due dispositivi a blocchi residenti su due nodi differenti attraverso una connessione di rete privata. Semanticamente il funzionamento è simile al RAID1 attuato però fra dispositivi remoti.

Tecnicamente DRBD è un modulo del kernel che mette a disposizione dei computer, tipicamente due, uno (o più) "block device" distribuito su entrambi i computer e quindi accessibile da entrambi. Ogni block device è formato da una o più partizioni residenti rispettivamente su entrambi i computer nelle quali i dati vengono replicati. In tal modo se una macchina cade, l'altra ha comunque la possibilità di accedere ai dati replicati sulla sua partizione e quindi continuare a lavorare. Le politiche di replicazione dei dati sono gestite secondo tre protocolli che bilanciano più o meno prestazioni e affidabilità. La scelta di tali protocolli va fatta in base alle esigenze che si hanno e all'hardware di cui si dispone.

Per quanto riguarda l'accesso ai dati consentito da DRBD, per garantire costantemente l'integrità dei dati, le macchine non possono effettuare contemporaneamente operazioni di scrittura. Vengono perciò stabiliti dei ruoli fra le due macchine: quella abilitata alla scrittura viene definita *primaria* mentre l'altra è detta *secondaria*. La macchina secondaria non può comandare operazioni di scrittura sui block device ma solo scrivere sulla propria partizione la replica dei dati al comando del nodo primario. L'algoritmo del DRBD gestisce tutte le situazioni in cui si verificano crash della macchina primaria o secondaria

¹<http://www.drbd.org>

con opportuno scambio dei ruoli, al fine di garantire sempre l'integrità dei dati e limitare al minimo l'intervento dell'amministratore, favorendo per quanto possibile una fornitura pressoché continua dei servizi offerti dalle macchine.

Heartbeat è il software che permette l'alta disponibilità. Attraverso questo software, i due server si controllano vicendevolmente e costantemente. Quando il master cade, lo slave si fa carico di tutti i servizi erogati da quest'ultimo. Per la migrazione degli indirizzi viene eseguito un meccanismo detto *ARP*.

Capitolo 2

Architettura hardware, versioni del software e accessori

- hardware Per completezza le macchine usate per questa installazione sono due workstation HP xw4300 con, ciascuna, 1GB di RAM, 2 dischi SATA da 160GB, 3 schede di rete (una Broadcom NetXtreme BCM5752 da 1GB integrata e due Realtek RTL-8139 da 100M aggiuntive).
- software Le versioni del software utilizzato sono il LiveCD Gentoo 2005.1, raidtools 1.00.3-r6, drbd 0.7.11, heartbeat 1.2.3-r1, e kernel vanilla 2.6.14.2.
- accessori Occorreranno un paio di cavi di rete cross per il collegamento punto/punto delle due schede di rete aggiuntive ed eventualmente un cavo null modem per la connessione delle macchine via seriale.

Capitolo 3

Installazione di Gentoo

I passi per installare Gentoo sono i soliti descritti nel manuale, eccetto per la fase di configurazione e inizializzazione delle partizioni. I dischi SATA vengono visti come dischi SCSI e quindi i device usati saranno `/dev/sda` per il disco primario e `/dev/sdb` per il secondario.

Per questa installazione sono state definite le seguenti partizioni: ¹

Partizione	Punto di montaggio	Dimensione	Filesystem
<code>/dev/sda1</code>	<code>/boot</code>	100MB	ext3
<code>/dev/sda2</code>	swap	1024MB	swap
<code>/dev/sda3</code>	<code>/</code>	2.5GB	reiserfs
<code>/dev/sda5</code>	<code>/usr</code>	5.0GB	reiserfs
<code>/dev/sda6</code>	<code>/var</code>	5.0GB	reiserfs
<code>/dev/sda7</code>	<code>/tmp</code>	1.5GB	reiserfs
<code>/dev/sda8</code>	<code>/opt</code>	2.5GB	reiserfs
<code>/dev/sda9</code>	<code>/portage</code>	5.0GB	reiserfs
<code>/dev/sda10</code>	<code>/bmr</code>	134.0GB	reiserfs

Lo stesso partizionamento è stato replicato sull'altro disco (`/dev/sdb`).

3.1 Caricamento dei moduli

Per poter utilizzare il raid software è necessario caricare il modulo `md`:

```
# modprobe md
```

3.2 Setup delle partizioni

Per configurare le partizioni si usa `fdisk`. Non ci sono differenze sostanziali per la creazione delle partizioni eccetto che tutte, meno quelle di swap, devono essere di tipo `fd` (linux raid auto-detect).

¹Le partizioni 1,2 e 3 sono primarie, la 4 è un'estesa, le successive sono partizioni logiche interne all'estesa.

Una volta salvata la configurazione, il comando `fdisk -l /dev/sda` dovrebbe mostrare un'output come il seguente:

```
Disk /dev/sda: 160.0 GB, 160041885696 bytes
255 heads, 63 sectors/track, 19457 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	1	13	104391	fd	Linux raid autodetect
/dev/sda2		14	138	1004062+	82	Linux swap / Solaris
/dev/sda3		139	443	2449912+	fd	Linux raid autodetect
/dev/sda4		444	19457	152729955	5	Extended
/dev/sda5		444	1052	4891761	fd	Linux raid autodetect
/dev/sda6		1053	1661	4891761	fd	Linux raid autodetect
/dev/sda7		1662	1844	1469916	fd	Linux raid autodetect
/dev/sda8		1845	2149	2449881	fd	Linux raid autodetect
/dev/sda9		2150	2758	4891761	fd	Linux raid autodetect
/dev/sda10		2759	19457	134134686	fd	Linux raid autodetect

3.3 Il file raidtab

Prima di creare qualsiasi filesystem, occorre creare e far partire il RAID. Per questo occorre creare il file `/etc/raidtab`. Questo file definisce come i drive virtuali RAID vengono mappati alle partizioni fisiche. Ecco un esempio:

```
# /boot (RAID 1)
raiddev      /dev/md0
raid-level   1
nr-raid-disks 2
chunk-size   32
persistent-superblock 1
device       /dev/sda1
raid-disk    0
device       /dev/sdb1
raid-disk    1

# / (RAID 1)
raiddev      /dev/md1
raid-level   1
nr-raid-disks 2
chunk-size   32
persistent-superblock 1
device       /dev/sda3
raid-disk    0
device       /dev/sdb3
raid-disk    1
```

```

# /usr (RAID 1)
raiddev      /dev/md2
raid-level   1
nr-raid-disks 2
chunk-size   32
persistent-superblock 1
device       /dev/sda5
raid-disk    0
device       /dev/sdb5
raid-disk    1

# /var (RAID 1)
raiddev      /dev/md3
raid-level   1
nr-raid-disks 2
chunk-size   32
persistent-superblock 1
device       /dev/sda6
raid-disk    0
device       /dev/sdb6
raid-disk    1

```

Il file conterrà tutte le partizioni create precedentemente. A questo punto si usa il comando **mkraid** per creare i device.

```
# mkraid /dev/md0
```

Se si riceve un errore come il seguente:

```
cannot determine md version: no MD device file in /dev
```

occorre eseguire il seguente comando:

```
# cd /dev ; MAKEDEV md
```

Questo comando crea tutti i file */dev/md** necessari. Riprovare quindi ad eseguire **mkraid** su tutti device raid presenti nel file */etc/raidtab*.

3.4 Creazione dei filesystem

Si possono ora creare i filesystem sui device raid.

IMPORTANTE: in questa guida si userà la partizione */dev/md7* (che corrisponde alle partizioni */dev/sda10* e */dev/sdb10*) per il DRBD. Sarà questa la partizione che verrà replicata fra le due macchine. La configurazione del DRBD su RAID software prevede i seguenti passi:

1. creazione del device RAID (con **mkraid**);
2. creazione del device DRBD sul RAID;
3. creazione del filesystem sul device DRBD;

Non sarà possibile creare il device DRBD su una partizione RAID sulla quale è stato creato un filesystem.

Detto questo, si creino i filesystem:

```
# mkswap /dev/sda2
# mkswap /dev/sdb2
# mke2fs -j /dev/md0
# mkreiserfs /dev/md1 ed a seguire tutte le partizioni meno l'ultima
```

3.5 Mount delle partizioni

Una volta creati i filesystem, si possono montare e si può proseguire l'installazione:

```
# swapon /dev/sda2
# swapon /dev/sdb2
# mount /dev/md1 /mnt/gentoo
```

Se si sono create tutte le partizioni come mostrato in questa guida occorrerà creare i punti di montaggio in */mnt/gentoo* e montare successivamente i filesystem:

```
# mkdir /mnt/gentoo/boot
# mkdir /mnt/gentoo/usr
.
.
# mount /dev/md0 /mnt/gentoo/boot
# mount /dev/md2 /mnt/gentoo/usr
.
.
```

Si copi quindi il file */etc/raidtab* nella nuova root di gentoo:

```
# cp /etc/raidtab /mnt/gentoo/etc/raidtab
```

Si prosegua quindi l'installazione come da manuale con l'estrazione dello *stage3* e dello *snapshot* del portage.

Una nota particolare in relazione col *portage*. Se si è creata una partizione specifica per il portage (come è stato fatto in questa guida), assicurarsi che il comando **tar** termini con l'opzione **-C /mnt/gentoo**.

3.6. NOTE FINALI: KERNEL, FSTAB, GRUB, PACCHETTI AGGIUNTIVIE RETE15

```
# tar jxvf /mnt/cdrom/snapshot/portage* -C /mnt/gentoo
```

Inoltre, dopo aver eseguito il **chroot**, modificare il file `/etc/make.conf` affinché la variabile `PORTDIR` punti a `/portage` oppure creare un link simbolico `ln -s /portage /usr/portage`.

3.6 Note finali: kernel, fstab, grub, pacchetti aggiuntivi e rete

- kernel

Durante la configurazione del kernel ricordarsi di abilitare il supporto per il RAID e per il device mapper. Assicurarsi di compilare il supporto nel kernel invece che come modulo, altrimenti il modulo deve venire caricato prima di montare i device RAID.

```
Device Drivers --->
  Multi-device support (RAID and LVM) --->
    [*] Multiple devices driver support (RAID and LVM)
    <*> RAID support
    <*> RAID-1 (mirroring) mode
    <*> Device mapper support
```

Abilitare anche il supporto per il *bonding* che servirà per il DRBD:

```
Device Drivers --->
  Network device support --->
    <M> Bonding driver support
```

- fstab

Ricordarsi di specificare i drive RAID nel file `/etc/fstab`:

```
/dev/md0      /boot        ext3    noauto,noatime 1 2
/dev/md1      /            reiserfs    noatime          0 1
/dev/md2      /usr         reiserfs    noatime          0 1
/dev/md3      /var         reiserfs    noatime          0 1
/dev/md4      /tmp         reiserfs    noatime,noexec,nosuid,no
dev 0 1
/dev/md5      /opt         reiserfs    noatime          0 1
/dev/md6      /portage     reiserfs    noatime          0 1
/dev/sda2     none         swap        sw                0 0
/dev/sdb2     none         swap        sw                0 0
```

Si noti che non si è inclusa l'ultima partizione (*/dev/md7*) riservata al DRBD.

- grub

La configurazione del **grub** avviene normalmente eccetto che nel file */boot/grub/menu.lst*, la voce *root* nella riga del kernel deve puntare al device RAID:

```
kernel (hd0,0)/vmlinuz root=/dev/md1
```

- pacchetti aggiuntivi

Installare i pacchetti *raidtools* e *reiserfsprogs* se si è optato per questo tipo di filesystem:

```
# emerge reiserfsprogs && emerge raidtools
```

- rete

Le schede di rete *eth1* e *eth2* di entrambe le macchina dovranno essere collegate con cavi cross, potranno quindi essere configurate in un secondo momento. Configurare solo la scheda *eth0*.

Riavviare la macchina.

Capitolo 4

Replicare l'installazione

Se il primo boot a Gentoo va a buon fine, si possono installare i pacchetti aggiuntivi quali *vim*, un'interfaccia grafica (*gnome*, *xfce*, *kde*, ecc), *gentoolkit*, ecc.

Quando tutto funziona correttamente, si può replicare l'installazione sulla seconda macchina in modo tale da avere alla fine due macchine identiche.

Capitolo 5

DRBD: preliminari

5.1 Installazione del channel-bonding

5.1.1 Introduzione

Le macchine hanno due schede di rete aggiuntive identiche per la comunicazione di sincronizzazione interna. Si utilizzeranno due schede per evitare che l'utilizzo di una sola rappresenti un "single point of failure".

Il channel-bonding è una funzionalità fornita con i sorgenti del kernel la cui implementazione richiede una corretta configurazione del kernel utilizzato (come visto in precedenza) e la compilazione del programma necessario a rendere effettivo questo tipo di connessione.

In pratica è possibile configurare il channel-bonding in modo tale che le due interfacce di rete siano utilizzate contemporaneamente per incrementare le performance e l'affidabilità. In questo modo infatti, venendo meno una, l'altra continuerebbe a connettere i due computer. Le due interfacce possono essere configurate per lavorare singolarmente in modo tale che un'interfaccia rimanga in stand-by e venga attivata nel caso in cui quella attiva dovesse rendersi non disponibile.

5.1.2 Installazione

All'interno dei sorgenti del kernel in `/usr/src/linux/Documentation/network` è collocato il file `ifenslave.c` che realizza effettivamente il bonding fra le due interfacce di rete.

Per effettuare il bonding perciò è necessario compilare il programma `ifenslave.c`.

```
# cd /usr/src/linux/Documentation/network
# gcc -Wall -Wstrict-prototype -O -I/usr/src/linux/include ifenslave.c -o ifenslave
```

Il file ottenuto lo si deve copiare nella directory degli eseguibili di sistema con

`cp ifenslave /sbin`. Si dà per scontato che `/usr/src/linux` sia un collegamento alla directory contenente i sorgenti del kernel che si sta usando.

La configurazione si completa inserendo nel file `/etc/modules.conf` la riga

```
alias bond0 bonding mode=0 miimon=100
```

Per attivare il channel-bonding è sufficiente caricare il modulo `bonding` e configurare l'interfaccia di rete astratta `bond0` con:

```
# modprobe bonding
# ifconfig bond0 192.168.0.1 up
```

e aggiungere al meccanismo di bonding le interfacce di rete reali che nel caso di questa guida sono `eth1` e `eth2`

```
# ifenslave bond0 eth1 eth2
```

Tale configurazione va ripetuta allo stesso modo anche sulla seconda macchina con la sola accortezza di cambiare nel comando `ifconfig bond0 192.168.0.1` l'indirizzo IP con `192.168.0.2`.

Si possono porre tutti comandi nel file `/etc/conf.d/local.start` per farli eseguire ad ogni riavvio, ed il modulo in `/etc/modules.autoload.d/kernel2.6` affinché venga caricato all'avvio.

```
# vi /etc/conf.d/local.start
ifconfig bond0 192,168.0.1 up
ifenslave bond0 eth1 eth2
```

```
# echo bonding >> /etc/modules.autoload.d/kernel2.6
```

Le opzioni inserite nel file `/etc/modules.conf` stanno ad indicare il particolare uso che si vuole fare del channel-bonding:

`mode=0` specifica al driver di usare alternativamente le schede di rete in base alla disponibilità ad accettare dati da trasportare secondo l'algoritmo del Round Robin;

`miimon=100` indica l'attivazione della funzione di monitoraggio del collegamento su ogni interfaccia di rete coinvolta nel bonding in modo tale che se su un'interfaccia il collegamento è caduto, non saranno inviati pacchetti che poi andrebbero persi. La funzionalità di monitoring deve essere supportata dalle interfacce di rete e il valore 100 indica l'intervallo di tempo in millisecondi fra un controllo e il successivo.

L'output del comando **ifconfig** dopo aver configurato e inizializzato correttamente il channel-bonding è il seguente:

```
bond0    Link encap:Ethernet  HWaddr 00:02:44:13:0D:19
         inet addr:192.168.0.1  Bcast:192.168.0.255  Mask:255.255.255.0
         UP BROADCAST RUNNING MASTER MULTICAST  MTU:1500  Metric:1
         RX packets:59205586  errors:0  dropped:0  overruns:0  frame:0
```

```

TX packets:102062603 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:1767467888 (1685.5 Mb) TX bytes:3066021466 (2923.9 Mb)

eth1    Link encap:Ethernet  HWaddr 00:02:44:13:0D:19
        UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500  Metric:1
        RX packets:59205586 errors:0 dropped:0 overruns:0 frame:0
        TX packets:102062603 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:1767467888 (1685.5 Mb) TX bytes:3066021466 (2923.9 Mb)
        Interrupt:20 Base address:0x1000

eth2    Link encap:Ethernet  HWaddr 00:02:44:13:0D:19
        UP BROADCAST SLAVE MULTICAST  MTU:1500  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)
        Interrupt:21 Base address:0x1100

```

Si noti che le interfacce di rete reali hanno un MAC-address identico, lo stesso che viene impostato al dispositivo virtuale bond0 e determinato dalla prima interfaccia di rete.

Testare la configurazione con un ping tra le due macchine.

5.1.3 Installazione di pconsole

Uno dei programmi sicuramente più utilizzati in ambito cluster è **pconsole**. Si noti che per il suo utilizzo occorre avere installato un'interfaccia grafica.

Tale software ha la funzione di replicare i comandi impartiti da console a tutti i computer che si desidera attraverso una connessione *ssh*.

È utile installare tale pacchetto dato che i comandi per l'installazione e la configurazione del DRBD dovranno essere forniti ad entrambe le macchine.

```
# emerge pconsole
```

È possibile che il pacchetto sia mascherato nel qual caso far precedere il comando precedente da *ACCEPT_KEYWORDS=~x86*.

Al momento, il pacchetto pconsole-1.0-r1 introduce un errore in **pconsole.sh** configurando la variabile *prefix* con un path non corretto. Per cui utilizzando il comando:

```
# pconsole.sh 192.168.0.1 192.168.0.2
```

si aprono due console con l'errore:

```
pconsole: Can't execvp
/var/tmp/portage/pconsole-1.0-r1/image//usr/bin/pconsole: No such file or
directory
```

Per risolvere il problema, editare `/usr/bin/pconsole.sh` e modificare la variabile `prefix` come segue:

```
prefix=/usr
```

5.2 Installazione del DRBD

Per facilitare le successive operazioni, utilizzare `pconsole.sh` per poter eseguire i comandi contemporaneamente sulle due macchine.

```
# pconsole.sh 192.168.0.1 192.168.0.2
```

Si apriranno tre finestre: due sono i terminali connessi alle due macchine mentre la terza è la console attraverso la quale digitare i comandi che appariranno in contemporanea sui terminali. Inserire la password di root nei terminali ed utilizzare la console per digitare i comandi per proseguire l'installazione di DRBD.

Per il seguito dell'installazione del DRBD i comandi dati attraverso pconsole saranno preceduti dal simbolo `>`, mentre i comandi eseguiti su ogni singola macchina saranno preceduti dal nome della macchina nella forma `master#` o `slave#`. Se non si usa pconsole, i comandi dovranno essere eseguiti su entrambe la macchine.

Gentoo fornisce l'ebuild per il DRBD e quindi per installarlo basta il comando:

```
> emerge drbd
```

Il pacchetto DRBD fornisce due comandi (`drbdsetup` e `drbdadm`) ed un modulo per il kernel.

Il caricamento del modulo con `modprobe drbd` inizializza due device: `/dev/drbd0` e `/dev/drbd1`.

Si noti che a partire dalla versione 0.7 di DRBD, non si usano più i device `/dev/nbd/X` ma sono stati ufficializzati i due device suddetti che le moderne distribuzioni supportano completamente.

5.3 Configurazione del DRBD

Per configurare il DRBD occorre creare il file `/etc/drbd.conf` (se ne può trovare un esempio in `/usr/share/doc/drbd-0.7.11/`). col seguente contenuto:

```
resource drbd0 {
    protocol C;
    syncer {
```

```

    group 0;
    rate 5M;
  }

on master {
    device /dev/drbd0;
    disk /dev/md7;
    address 192.168.0.1:7788;
    meta-disk internal;
  }
on slave {
    device /dev/drbd0;
    disk /dev/md7;
    address 192.168.0.2:7788;
    meta-disk internal;
  }
}

```

Modificare *master* e *slave* coi nomi dati alle due macchine.
Eseguire il seguente comando

```
> drbdadm up all
```

Questo dovrebbe inizializzare entrambe le macchine nello stato *Secondary* e *Inconsistent*.

Questo stato lo si può evincere usando **dmesg** | **tail** e **cat /proc/drbd**.

A questo punto occorre scegliere la macchina primaria dando il seguente comando sul terminale della macchina prescelta:

```
master# drbdadm primary all
```

l'output del comando dovrebbe assomigliare al seguente:

```

ioctl(,SET_STATE,) failed: Input/output error
Local replica is inconsistent (--do-what-I-say ?)
Command line was '/sbin/drbdsetup /dev/drbd0 primary'
drbdsetup exited with code 21

```

Questo messaggio è quello che ci si aspetta. Dopo un ulteriore controllo per assicurarsi di essere sulla macchina master, si esegue il seguente comando che forza la macchina ad essere la primaria:

```
master# drbdadm -- --do-what-I-say primary all
```

Se il comando precedente non dà errori, si può controllare in */proc/drbd* che sia iniziata la sincronizzazione tra le due macchine.

```
master# cat /proc/drbd
```

```
version: 0.7.11 (api:77/proto:74)
SVN Revision: 1807 build by root@icnm, 2005-12-20 03:54:50
0: cs:SyncSource st:Primary/Secondary ld:Consistent
   ns:12940824 nr:0 dw:87492 dr:13690591 al:109 bm:1668 lo:1000 pe:1876 ua:1000 ap
   [=====>.....] sync'ed: 44.4% (15858/28487)M
   finish: 0:09:20 speed: 28,933 (25,160) K/sec
```

Si può creare adesso il filesystem sul device DRBD solo sulla macchina primaria:

```
master# mkreiserfs /dev/drbd0
```

Si può testare se tutto funziona anche in fase di sincronizzazione:

```
master# mount /dev/drbd0 /bmr && touch /bmr/test
master# umount /bmr && drbdadm secondary all
```

```
slave# drbdadm primary all
slave# mount /dev/drbd0 /bmr && ls -l /bmr/test
```

I comandi dati montano il device DRBD e creano un file vuoto sul master. Quindi di rende il master secondario, il secondario diviene primario per permettere il montaggio del device DRBD e il check sull'esistenza del file di test.

Si riporta, quindi, tutto all'origine.

```
slave# umount /bmr
slave# drbdadm secondary all
```

```
master# drbdadm primary all
```

Si può ora inserire la seguente voce nel file */etc/fstab* per il device DRBD:

```
/dev/drbd0 /bmr reiserfs noauto 0 0
```

Capitolo 6

Heartbeat

6.1 Preliminari

Prima di immergersi nell'installazione e configurazione di Heartbeat, occorre avere ben chiaro quali dovranno essere i servizi che dovranno essere erogati dal master e quindi fatti partire sullo slave nel caso il master si fermi.

Nel nostro caso, i servizi gestiti da Heartbeat saranno Apache, MySQL e DRBD.

Un'altra chiarificazione riguarda gli indirizzi IP pubblici che entreranno in gioco. Le due macchine dovranno essere configurate per avere le due interfacce di rete principali connesse ad Internet con indirizzi IP pubblici (a meno che il cluster non sia posto dietro ad un firewall, ma non è il caso preso in esame). Nessuno dei due è però l'indirizzo IP pubblico attraverso il quale i client usufruiranno dei servizi e che di solito è stato registrato sul server DNS come `www.miodominio.it` o `mail.miodominio.it`.

Per esempio, se sul DNS è stato registrato `www.miodominio.it` con indirizzo IP `10.10.10.1` (non è un indirizzo pubblico, ma è solo per esempio), il suddetto indirizzo non dovrà essere assegnato a nessuna delle due macchine, ma verrà gestito da Heartbeat.

Per evitare un Single point of failure, sarebbe opportuno collegare le due macchine anche via seriale con un cavo null modem.

6.1.1 Resource Group

Nella logica di Heartbeat si definisce risorsa ogni entità del sistema che Heartbeat gestisce. Una risorsa può essere ad esempio un indirizzo IP, un servizio come Apache, un componente hardware come una unità disco, una risorsa del kernel come un filesystem.

Nel momento in cui il master cade, il service group viene interamente migrato sullo slave e le risorse così trasferite vengono attivate così com'erano sulla macchina caduta.

6.2 Installazione

Utilizzando sempre pconsole si installa heartbeat su entrambe le macchine:

```
> emerge heartbeat
```

6.3 Configurazione

I file di configurazione sono in */etc/ha.d*, ma quelli necessari sono sostanzialmente tre: *ha.cf*, *haresources* e *authkeys*.

La configurazione dei tre file elencati evidenzia le politiche di gestione delle risorse che si sono decise in fase di progetto. Da tenere presente che i file di configurazione sono identici su entrambe le macchine.

6.3.1 Il file haresources

All'interno del file *haresources* vengono configurati i Resource Group.

Le righe all'interno del file sono simili alle seguenti:

```
master drbddisk::drbd0 Filesystem::/dev/drbd0::/bmr::reiserfs \
193.206.185.XXX apache2 mysql
```

All'inizio della riga c'è l'hostname della macchina master la quale eroga il servizio (come specificato dall'output del comando *uname -n*). Per i servizi che si devono erogare è stato configurato un device DRBD denominato *drbd0* (come da file */etc/drbd.conf* dopo la voce *resource*). Attraverso lo script *drbddisk* Heartbeat è in grado di dare alla macchina il ruolo di primario per il device specificato provvedendo inoltre a montare il device (*/dev/drbd0*) al percorso specificato (*/bmr*). Per questo motivo nel file */etc/fstab* il dispositivo DRBD deve essere configurato con l'opzione *noauto* per far sì che venga montato correttamente da Heartbeat. La specifica del *Filesystem* termina con il tipo di filesystem creato sul dispositivo DRBD. Segue l'indirizzo 193.206.185.XXX che è l'indirizzo IP gestito da Heartbeat (attraverso ARP), che non deve essere assegnato a nessuna delle due macchine e che, essendo registrato sul DNS, è l'indirizzo attraverso il quale viene erogato il servizio ai client.

Terminano la riga i servizi che si intendono far gestire da Heartbeat e che verranno avviati sul master o sullo slave nel caso il primo cada. Il nome di tali servizi fanno riferimento ai relativi file di start/stop in */etc/init.d*.

6.3.2 Il file ha.cf

Il file *ha.cf* viene letto da Heartbeat nel momento in cui viene lanciato attraverso il comando

```
# /etc/init.d/heartbeat start
```

Tale file contiene le impostazioni necessarie affinché la suite sia configurata per adattarsi alle specifiche del sistema su cui va ad essere eseguita per soddisfare le politiche di gestione che si sono stabilite in fase di progetto.

Segue un'analisi del file con le opzioni più importanti:

- *keepalive2* - specifica il tempo in secondi entro cui inviare due segnali di heartbeat fra le due macchine
- *deadtime 60* - stabilisce il tempo in secondi entro cui non avendo risposto al segnale di heartbeat, una macchina può essere considerata funzionante
- *initdead 120* - imposta a 120 secondi il tempo di *deadtime* solo per la prima volta, quindi subito dopo l'avvio di Heartbeat per far in modo che si consideri il tempo necessario a stabilire la comunicazione fra le macchine e ad inizializzare i diversi plugin che gestiscono le diverse funzioni di Heartbeat
- *serial /dev/ttyS0* - definisce la porta seriale attraverso la quale inviare i segnali di heartbeat
- *auto_failback on* - sta ad indicare un particolare comportamento del cluster. Secondo l'impostazione applicata quando una macchina rientra nel cluster dopo esserne stata esclusa per malfunzionamento o per manutenzione, in quest'ultimo caso volontariamente da parte dell'amministratore, il service group configurato per default su tale macchina viene automaticamente riportata su di essa.

E' evidente che i servizi torneranno ad essere erogati su tale macchina solo dopo che il filesystem condiviso è stato sincronizzato. Il compito di verificare la sincronizzazione dei block-device condivisi è assolto da DRBD.

- *ucast bond0 192.168.0.2* - specifica gli altri canali utilizzati per il segnale di sincronizzazione ridondando quello seriale ed evitando così l'introduzione di Single point of failure. In questo caso il canale è l'interfaccia di bonding.
- *node newicnm*
- *node newicnm2* - si definiscono i nomi delle due macchine facenti parte il cluster.
- *respawn root /etc/init.d/apache2*
- *respawn root /etc/init.d/mysql* - si definiscono gli script di avvio dei servizi da erogare come definiti in *haresources*. Questi vengono avviati come utente root. Il comando *respawn* ad inizio riga, specifica che Heartbeat si preoccuperà di monitorare costantemente il funzionamento del processo e di rilanciarlo nel caso in cui venga interrotto.

6.3.3 Il file `authkeys`

Tutte le comunicazioni che avvengono fra i nodi sono cifrate e autenticate per evitare problemi di sicurezza che potrebbero falsare il corretto funzionamento di Heartbeat.

Attraverso i plugin di autenticazione, Heartbeat consente l'utilizzo di algoritmi crittografici più o meno sicuri affinché si renda possibile un corretto bilanciamento fra prestazioni e sicurezza in relazione all'ambiente operativo del cluster.

Nel file `authkeys` si configura appunto l'algoritmo crittografico per le comunicazioni fra le macchine.

Un file tipo con le schede di rete collegate via cross potrebbe essere:

```
auth 3
3 md5 miachiave
```

Dove viene definito il metodo di crittografia md5 seguito dalla chiave da utilizzare per la comunicazione.

6.3.4 Partenza di Heartbeat

Ci si ricordi che i file di configurazione di Heartbeat presi in esame nella sezione precedente devono essere gli stessi su entrambe le macchine.

A questo punto si è pronti per far partire Heartbeat. Naturalmente nessuno dei servizi gestiti da Heartbeat (Apache e MySQL nel nostro esempio) dovranno essere posti nel run level di default per la partenza automatica ad ogni riavvio delle macchine.

Controllare con `rc-update show` che i servizi non siano attivi.

Si faccia partire Heartbeat:

```
# /etc/init.d/heartbeat start
```

Con `ps aux` si può verificare la partenza dei servizi ed il mount della partizione gestita da DRBD altrimenti controllare `/var/log/ha-log` per la presenza di errori.

6.3.5 Test

I test sono il punto fondamentale di una configurazione del genere. Alcuni test da fare riguardano il bonding. Avviare un ping attraverso l'interfaccia di bonding, staccare un cavo di rete che collega le due interfacce facenti parte del bonding, questo non si deve fermare.

Fermare la macchina master. Dopo qualche secondo sullo slave dovrebbero partire i servizi, si dovrebbe avere la partizione gestita da DRBD montata, e guardando `/proc/drbd` si dovrebbe avere uno stato Primary/Unknown. Fare ripartire il master. Una volta terminato il caricamento del sistema operativo si dovrebbero riavere i servizi attivi su tale macchina e non sulla macchina

slave. In `/proc/drbd` la situazione dovrebbe essere tornata nello stato Primary/Secondary. Se durante lo spegnimento della macchina master, si sono creati file sulla macchina slave nella partizione gestita da DRBD, tali file si dovrebbero ripresentare anche sulla macchina master una volta riavviata. Un ping all'indirizzo pubblico gestito da Heartbeat non dovrebbe fermarsi né durante lo spegnimento della macchina master né durante la sua riaccensione. Se da un client si sta interrogando il server Apache attraverso l'indirizzo IP gestito da Heartbeat, non ci dovrebbero essere tempi morti nel caricamento di nuove pagine web durante lo spegnimento del master, se non di qualche secondo. Stessa cosa durante la sua riaccensione.

Capitolo 7

Troubleshooting

7.1 RAID e DRBD

Se non si sono seguite attentamente le raccomandazioni di non creare un filesystem nella partizione riservata al DRBD, il comando **drbdadm up all** darà un errore che fa riferimento ad un device busy o in uso.

Per ovviare a questo problema si seguano questi passi:

- si utilizzi **fdisk** per rimuovere le partizioni */dev/sda10* e */dev/sdb10* e ricrearle di tipo linux
- rimuovere le voci che fanno riferimento a queste partizioni da */etc/raidtab*
- riavviare la macchina
- riutilizzare **fdisk** per cambiare il tipo delle partizioni in oggetto per renderle nuovamente RAID (tipo *fd*)
- reinserire le voci nel file */etc/raidtab*
- bloccare il raid sul device con **raidstop /dev/md7**
- creare nuovamente il device raid con **mkraid -R /dev/md7**

provare a ricreare il device DRBD.

7.2 Heartbeat e DRBD

Se testando la configurazione, una delle macchine si riavvia e lo stato del DRBD è standalone, c'è probabilmente il problema che il timeout di Heartbeat è più corto di quello del DRBD.

Incrementare il *deadtime* di Heartbeat per essere maggiore del timeout del DRBD. Per questo impostare *deadtime 120* in *ha.cf* e aggiungere una sezione in *drbd.conf* come la seguente:

```
net {  
    timeout 30;  
}
```