HA (high availability) Webserver mit Drbd und heartbeat

Server sollten in einem Netzwerk ständig verfügbar sein. Der Ausfall eines Servers kann zur Arbeitsunfähigkeit führen und den ganzen Betrieb stilllegen.
Um solch ein Szenario zu vermeiden lässt sich mit recht einfachen Mitteln ein System bauen, in dem ein Reserve-Server einspringt wenn einmal der Hauptserver den Dienst versagt.

Ansätze dazu gibt es viele, ich möchte 2 gleichberechtigte **APACHE2** Webserver mit **MYSQL** und **PHP** so konfigurieren, dass ***Server_B*** die Aufgabe des ***Server_A*** übernimmt, sobald ***Server_A*** ausfällt. Nachdem ***Server_A*** wieder verfügbar ist, soll nun wiederum er als Reserve fungieren und erst beim Ausfall des nun aktiven ***Server_B*** wieder die Rolle des aktiven Server übernehmen. Es findet also gewisser Massen ein Rollentausch zwischen ***Server_A*** und ***Server_B*** statt. Wichtig ist mir, dass alle Einstellungen des Webservers und der Datenbanken aktuell bleiben. Wird also an der Konfiguration des Servers etwas geändert, soll der Ersatz-Server diese Einstellungen automatisch übernehmen. System-Updates sollen allerdings davon nicht betroffen sein, da ein fehlerhaftes Update, das S***erver_A*** stilllegt auch ***Server_B*** betreffen würde.

Als Basis-System dienen 2 Debian (Squeeze) Server ohne Grafische Oberfläche, die zusätzlich mit den Paketen drbd8-utils und heartbeat versehen werden.

drbd8-utils

Drbd8-utils ist ein Tool zum Erstellen eines Raid1 Festplatten Verbundes über eine Netzwerkkarte. Dabei werden mehrere Festplatten logisch zu einer Platte zusammengefasst, die physikalisch an unterschiedlichen Orten liegen (bei 2 Festplatten liegt eine im ***Server_A*** und eine im ***Server_B***). Bei einem RAID-System werden alle Platten synchron gehalten und verfügen immer über die Daten der anderen. Lege ich also Daten auf ***Server_A*** ab, sorgt das RAID-System dafür, dass die Daten auch auf ***Server_B*** abgelegt werden. Drbd geht dabei immer von einem aktiven und einem passiven Gerät aus und synchronisiert das aktive auf das passive.

heartbeat

Heartbeat ist eine Netzwerkverbindung zwischen zwei (oder mehr) Rechnern, um sich gegenseitig darüber zu benachrichtigen, dass sie betriebsbereit sind und ihre Aufgaben noch erfüllen können. Bleibt diese Benachrichtigung aus, kann Heartbeat die Aufgabe des „ausgefallenen“ Rechners an einen verbleibenden Rechner übertragen.
Die Fähigkeit des Dienstes virtuelle IP-Adressen vergeben zu können und Devices ins System einbinden zu können, sowie den Start von Servern (Diensten) zu managen, macht heartbeat zu einem idealen Tool um unsere Webserver-Aufgaben zu verwalten.

Die Hardware

Zum Einsatz kommen 2 Rechner mit jeweils

- 2 x 250 GB Systemplatten (als RAID eingerichtet)
- 2 x 500 GB Datenplatten2
- 2 Gib Netzwerkkarten

Die Datenplatten (Partitionen) sollten die gleiche Größe haben, da sie mit Drbd zu einem RAID zusammengefasst werden.

Der Aufbau in der Übersicht (Abstrakt)

Ich gehe von 2 Servern aus, auf denen bereits ein funktionierendes Debian mit Apache2, PHP und Mysql läuft.

Als ersten Schritt werden wir mit Drbd eine der Datenplatten von ***Server_A*** und eine von ***Server_B*** zu einem Raidsystem verbinden. Unser RAID ist also auf 2 Rechner verteilt. Mit Drbd werden wir auf beiden Servern ein Device erstellen, über das wir auf dieses Raidsystem zugreifen können. Danach werden wir ***Server_A*** zum aktiven Server machen und das drbd-Device auf diesem Rechner einbinden. Auf diese Drbd-Ressource werden wir den Apache2 Webserver, PHP und MYSQL verschieben. Da unser Device ein Raidsystem ist, wird jede Aktivität auf dem aktiven Server automatisch auf den passiven Server synchronisiert.

Als zweiten Schritt werden wir heartbeat dazu bringen, zu überprüfen, ob der aktive ***Server_A*** noch im Netz ist. Wenn nicht wird heartbeat alle Aufgaben an Server_B übergeben, die Drbd-Ressource auf dem passiven Server einbinden, und alle notwendigen Dienste (Apache, MYSQL) starten. Da ein Webserver immer eine statische IP-Adresse hat, über die er erreichbar ist, wird heartbeat außerdem diese IP-Adresse von ***Server_A*** auf ***Server_B*** übertragen.

Im dritten und letzten Schritt werden wir auf der verbleibenden zweiten Datenplatte ein Backup einrichten, dass immer nur den aktiven Server sichern wird. Die Lösung ist ein Kompromiss, da (je nachdem welcher Server gerade der aktive ist) unser Backup auf 2 Rechnern verteilt sein wird, aber da ich das Backup in einem Logfile protokolliere, sollte es ein sekundäres Problem sein herauszufinden, welcher Server das letzte Backup erstellt hat, oder besser auf welchem Rechner das aktuellste Backup liegt.

Im gesamten Howto werden 2 Server verwendet, die ich in der Folge **www4** und **www5** nenne

Meine Partitionen sehen auf beiden Rechnern wie folgt aus, auf sdc ist das System installiert. Sdc besteht aus 2 250GB Platten, die in einem Raidverbund laufen.

```
   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1   *           1       29563   237463552   83  Linux
/dev/sdc2           29564       30319     6065153    5  Extended
/dev/sdc5           29564       30319     6065152   82  Linux swap / Solaris

Disk /dev/sdb: 500.1 GB, 500107862016 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1       32636   262148638+  83  Linux
/dev/sdb2           32637       60801   226235362+  83  Linux

Disk /dev/sda: 500.1 GB, 500107862016 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1               1       32636   262148638+  83  Linux
/dev/sda2           32637       60801   226235362+  83  Linux
```

Fangen wir also an.....

drbd8-utils

Auf beiden Servern wird Drbd mit

```
apt-get install drbd8-utils
```

installiert. Das Programm erlaubt es über eine Netzwerkverbindung ein Raidsystem zu erstellen und ein Device einzurichten, das diesen Raidverbund ansprechbar macht so dass man es ins System einbinden kann. Dabei betrachtet Drbd die einzelnen Rechner als eine Art Cluster-knoten.

Das Programm bekommt dafür exklusiv eine der beiden Netzwerkkarten unserer Server fest zugeordnet.

WICHTIG !

An dieser Netzwerkkarte dürfen nur Cluster-knoten angeschlossen sein, die auch in der Drbd-Konfiguration definiert sind. Ein Anschluss an einen Switch oder Router, an dem noch andere „Nicht-Clusterknoten“ angeschlossen sind ist damit ausgeschlossen. Ich verwende für meine 2 Server eine direkte Kabelverbindung (Krossover)

Die Netzwerkkarte wird in beiden Servern mit einer festen IP konfiguriert. (im Bsp eth1). Eth0 ist bei mir die Netzwerkkarte, an die der Webserver gebunden wird und mit der die restliche Kommunikation abgewickelt wird. Dazu später bei heartbeat.

Die Datei */etc/network/interfaces*

www4

```
auto eth1
iface eth1 inet static
        address 192.168.12.2
        netmask 255.255.255.0
        network 192.168.12.0
        broadcast 192.168.12.1
```

www5

```
auto eth1
iface eth1 inet static
        address 192.168.12.3
        netmask 255.255.255.0
        network 192.168.12.0
        broadcast 192.168.12.1
```

Damit sich die Knoten kennen editieren wir zunächst auf beiden Rechnern die Datei */etc/hosts* und tragen dort unsere Knoten ein.

```
127.0.0.1 localhost

192.168.12.3    www5.domain.tld   www5
192.168.12.2    www4.domain.tld   www4
```

Geben wir nun auf www4 auf der Konsole ein ping auf www5 ein, müsste eine Antwort erfolgen und umgekehrt.

Drbd wird im wesentlichen über eine Datei konfiguriert. */etc/drbd.conf*. Diese Datei sieht auf beiden Servern identisch aus.

Die Datei ist weitestgehend vorkonfiguriert, muss aber an einigen Stellen erweitert oder nachgebessert werden. Das Listing auf der Folgeseite zeigt eine Konfiguration für eine 100 Mb Netzwerkkarte .

Was welche Einstellung im einzelnen tut kann man den manpages b.z.w. http://www.drbd.org/ entnehmen. Eine kurze Beschreibung aber zu den Teilen *resource raid*, *on www4* und *on www5* (Listing siehe unten)

resource raid

Gibt unserer Raid Ressource einen Namen. Über diesen Namen kann mit den drbd-Tools (der Befehl lautet drbdadm) unser Raid angesprochen, verwaltet und manipuliert werden. Meine drbd Ressource heißt also **raid**.

```
on www4 { # um werlchen Rechner handelt es sich
        #  beschreibt den Device-Namen den ich zugeordnet habe.
  device     /dev/drbd0;

        #  auf welcher Festplatte/Partition soll die Ressource angelegt werden
  disk       /dev/sda1;

        #  unter welcher IP, also auf welchem Rechner wird diese Festplatte zu finden sein und
        #  welcher Port soll zur bildung des Raid verwendet werden
  address    192.168.12.2:7788;

        #  wo sollen Metadaten gespeichert werden.
        #  bei erweiterten Partitionnen wird hier eine primäre Partition als Speicherort
        #  angegeben.
  meta-disk  internal;
```

Wer eine Gib Netrzwerkkarte verwendet kann den Wert unter syncer „***rate 100M;***“ auch höher einstellen und damit einen höheren Durchsatz erzielen.

Das komplette Listing

```
# an Statistikauswertung teilnehmen
global {
    usage-count yes;
}

# Optionen für alle Ressourcen
 common {
  syncer {
    rate 100M;
  }
}

 resource raid {
 protocol C;

  startup {
    wfc-timeout 0;
    degr-wfc-timeout 120;    # 2 minutes.
  }

  disk {
    on-io-error   detach;
  }

  net {
    # sndbuf-size 512k;
    # timeout       60;    #  6 seconds  (unit = 0.1 seconds)
    # connect-int   10;    # 10 seconds  (unit = 1 second)
    # ping-int      10;    # 10 seconds  (unit = 1 second)
    # max-buffers     2048;
  }

  syncer {
    rate 100M;
  }

  on www4 {
    device     /dev/drbd0;
    disk       /dev/sda1;
    address    192.168.12.2:7788;
    meta-disk  internal;
  }



  on www5 {
    device    /dev/drbd0;
    disk      /dev/sda1;
    address   192.168.12.3:7788;
    meta-disk internal;
  }
}
```

Bitte darauf achten, dass die Datei auf beiden Rechnern identisch ist.

Das Folgende wird auf BEIDEN Rechnern ausgeführt!

Nun muss geprüft werden, dass die angegebene Partition nicht in der fstab eingetragen ist. Das Einbinden unserer Ressource soll später heartbeat übernehmen und soll nicht schon beim Start vom System erfolgen. War die Partition schon bei der Installation des Systems formatiert, steht sie wahrscheinlich in der fstab. Ich nutze auf beiden Rechnern /dev/sda1 als drbd Ressource. Um ein bestehendes System auf sda zu löschen kann folgender Befehl verwendet werden:

```
dd if=/dev/zero bs=10M count=100 of=/dev/sda1; sync
```

bs * count sollte dabei ca. der Grösse der Partition entsprechen, die von ihrem System befreit werden soll. Ist dieser Wert etwas grölsser gibt es zwar eine Meldung, das macht aber nichts.

Nun laden wir das Modul und legen einen Startlink an. Sollte hier eine Meldung erscheinen, dass der Link bereits existiert, ist das OK, da bei der Installation dieser Punkt eventuell schon erledigt wurde.

```
modprobe drbd
echo 'drbd' >> /etc/modules
update-rc.d drbd defaults
```

Wir sehen, ob das Modul geladen wurde:

```
lsmod | grep drbd
```

Der erste Start

Nun kann drbd zum ersten mal gestartet werden.

```
/etc/init.d/drbd restart
drbdadm create-md raid
drbdadm up raid
```

Drbd erstellt mit *create-md-raid* ein Device (/dev/drbd0), *drbdadm up raid* startet die Syncronisation.

Ob drbd läuft und syncronisiert kann mit einem einfachen cat-Befehl kontrolliert werden:

```
cat /proc/drbd
```

sollte ca. folgendes Bild liefern.

```
root@www4:/etc# cat /proc/drbd
version: 8.3.7 (api:88/proto:86-91)
srcversion: EE47D8BF18AC166BE219757
 0: cs:Connected ro:Primary/Secondary ds:UpToDate/UpToDate C r----
    ns:1109732 nr:1592 dw:1111324 dr:596887730 al:11770 bm:115 lo:0 pe:0 ua:0 ap:0 ep:1 wo:b oos:0
root@www4:/etc#
```

Sollten Fehlermeldungen sichtbar sein oder Meldungen wie *waiting for connect* sichtbar werden, könnten folgende Befehle helfen:

```
drbdadm disconnect raid
drbdadm detach raid
```

Danach werden die Punkte unter „der erste Start“ wiederholt.

Ab jetzt werden alle Schritte nicht mehr auf beiden Rechnern parallel ausgeführt, sondern nur auf dem jeweils angegebenen Rechner.

Primary oder Secondary

cat /proc/drbd leugnet die Konsistenz, das ist vollkommen normal, da wir noch nicht festgelegt haben, welcher der beiden Server der Primäre Server ist (es können ja nicht beide Webserver spielen). Das holen wir nun nach. Das ist wichtig, da drbd immer vom Primären Server auf den Sekundären Server synchronisiert. Ausserdem erlaubt drbd nur dem Primären Server eine Ressource zu mounten.

www4

```
drbdadm -- -o primary raid
```

Synchronisationsprozesse hinterlassen ihre Spuren in der Datei */var/log/syslog*, dort kann also kontrolliert werden, ob drbd läuft und funktioniert.

Als Abschluss erstellen wir nun ein Dateisystem auf unserer drbd-Ressource. Das muss nur auf dem Primary geschehen, da die Synchronisation dafür sorgt, dass die Ressource des Secondary mit dem Filesystem synchronisiert wird.

www4

```
mkfs.ext3 /dev/drbd0
```

Nachdem das Filesystem erstellt wurde kann unser Device das erste mal gemounted werden.

ACHTUNG !
Gebt drbd etwas Zeit, das Filesystem zu synchronisieren. Wer vor Abschluss der Synchronisation dir folgenden Befehle absetzt (also www4 zum Secondary macht) kann das raid beschädigen. Drbd reagiert etwas ungehalten, wenn während der Synchronisation Primary und Secondary wechseln. Es geht zwar nichts kaputt, aber drbd kommt etwas durcheinander. (siehe etwas weiter unten).

Ich habe mir auf beiden Servern ein Verzeichnis /srv/raid erstellt und dort mit einem chmod 777 die komplette Freigabe im System sichergestellt.

www4

```
mount /dev/drbd0 /srv/raid
```

Erstellen wir eine Datei und kontrollieren, ob drbd auch brav syncronisiert.

```
touch /srv/raid/test.txt
umount /srv/raid
drbdadm secondary raid
```

www5

```
drbdadm primary raid
mount /dev/drbd0 /srv/raid
ls -al
```

Die erstellt Datei test.txt sollte nun auf www5 zu sehen sein. Machen wir www4 wieder zum Primary!

www5

```
umount /srv/raid
drbdadm secondary raid
```

www4

```
drbdadm primary raid
mount /dev/drbd0 /srv/raid
```

Drbd läuft nun und die Installation ist damit abgeschlossen.

Anmerkung:

Mir ist aufgefallen, dass drbd8-utils etwas sensibel reagiert, wenn während eines Synchronisationsprozesses z.B. die Verbindung des Raid abbricht oder Primary und Secondary getauscht wird. Ich habe z.B. das Netzwerkkabel gezogen während Daten im Raid synchronisiert wurden. Danach ließ sich kein Primary mehr definieren und drbd hat aufgegeben. Ein mount schlug damit auch fehl und ich konnte auf meine Ressource nicht mehr zugreifen.

Lösung:

www4

```
drbdadm disconnect raid
drbdadm detach raid
drbdadm create-md raid
```

Die darauf folgende Frage von drbd ob das bestehende Device ersetzt werden soll habe ich mit Ja beantwortet.

```
drbdadm up raid
drbdadm — -o primary raid
```

hat www4 wieder zum Primär-Server gemacht und die Synchronisation neu gestartet.

Facit:

Wer grosse Datenmengen auf den Primären Server kopiert sollte unbedingt die komplette Synchronisation abwarten, bis er herumexperimentiert. Die Netzwerk-Synchronisation dauert naturgemäss etwas länger als über Bus oder Raidcontroller.

Drbd8-utils Befehle

```
drbdadm attach raid
```

Bindet eine Ressource neu ein und startet eine Neusynchronisation

```
drbdadm detach raid
```

Trennt die Verbindung zu einer Ressource

```
drbdadm primary/secondary raid
```

Definiert eine Ressource als Primary/Secondary

```
drbdadm role raid
```

Zeigt den Status (Primary/Secondary) einer Ressource an

```
drbdadm dstate raid
```

Zeigt den Staus der Synchronisation an

```
drbdadm invalide-remote raid
```

Daten des Partners als ungültig markieren und Synchronisation starten (nur Primary)

Verschieben von Apache, PHP undMysql

Www4 ist nun wieder der Primär-Server, /dev/drbd0/ ist gemountet ins Verzeichnis /srv/raid und wir können uns daran machen den Webserver, PHP und Mysql auf die drbd-Ressource zu verlegen.

www4

```
/etc/init.d/apache2 stop
/etc/init.d/mysql stop
mv /etc/apache2 /srv/raid/
ln -s /srv/raid/apache2 /etc/apache2/
mv /var/www/ /srv/raid/www/
ln -s /srv/raid/www /var/www/
mv /etc/php5/ /srv/raid/php5/
ln -s /srv/raid/php5 /etc/php5/
mv /var/lib/mysql/ /srv/raid/mysql/
ln -s /srv/raid/mysql/ /var/lib/mysql/
```

www5

```
mv /etc/apache2 /etc/apache2org/
ln -s /srv/raid/apache2 /etc/apache2/
mv /var/www/ /var/wwworg/
ln -s /srv/raid/www /var/www/
mv /etc/php5/ /etc/php5org/
ln -s /srv/raid/php5 /etc/php5/
mv /var/lib/mysql/ /var/lib/mysqlorg/
ln -s /srv/raid/mysql/ /var/lib/mysql/
```

Geschafft !!

Heartbeat

Der Dienst kann extrem viel, für unsere Zwecke nutzen wir 3 Fähigkeiten.

1. Automatisches Einbinden (mounten) von drbd-Ressourcen
2. Vergeben einer virtuellen IP
3. Starten von Diensten auf dem aktiven (primären) Server

Heartbeat kontrolliert, ob ein Clusternode seine Aufgaben erfüllen kann oder nicht. Wenn nicht, kann heartbeat die Aufgabe an einen anderen Cluster-knoten übergeben. In unserem Fall werden also die Dienste von Apache2 und Mysql an www5 übergeben, wenn www4 ausfällt. Die Kommunikation von heartbeat erfolgt über die erste Netzwerkkarte (eth0) unserer Server. In meinem Fall ist eth0 mit einer festen IP konfiguriert. Der Webserver hat ebenfalls eine feste IP, die aber NICHT in /etc/network/interfaces eingetragen ist da, je nachdem welcher Server gerade der aktive ist, die IP-Adresse von Machine zu Maschine wechselt. Heartbeat wird also die eine öffentliche Webserver-IP dynamisch an den Primärserver übergeben

Die öffentliche Webadresse wird im Beispiel die IP ***133.10.10.10*** sein

www4 */etc/network/interfaces*

```
iface eth0 inet static
    address 133.10.10.5
    netmask 255.255.255.0
    network 133.10.10.0
    broadcast 133.10.10.255
    gateway 133.10.10.1
    dns-nameservers 133.10.10 10.3
    dns-search domain.tld
```

www5 */etc/network/interfaces*

```
iface eth0 inet static
    address 133.10.10.6
    netmask 255.255.255.0
    network 133.10.10.0
    broadcast 133.10.10.255
    gateway 133.10.10.1
    dns-nameservers 133.10.10 10.3
    dns-search domain.tld
```

Heartbeats Konfiguration befindet sich in 3 Dateien.

/etc/heartbeat/ha.cf, */etc/heartbeat/haresources* und */etc/heartbeat/authkeys*.

www4 /etc/heartbeat/ha.cf

```
logfile /var/log/heartbeat.log
auto_failback   off
ucast     eth0  133.10.10.6
node      www4
node      www5
keepalive 2
deadtime  30
```

www5 /etc/heartbeat/ha.cf

```
logfile /var/log/heartbeat.log
auto_failback   off
ucast     eth0  133.10.10.5
node      www4
node      www5
keepalive 2
deadtime  30
```

Damit kann heartbeat nun kontrollieren, ob der Server auf der Gegenseite verfügbar ist oder nicht. Ich nutze ucast zum Testen, es kann aber ebenso bcast (Broadcast) verwendet werden. Alle Nodes die im Cluster vorhanden sind werden hier eingetragen, nicht eingetragene Nodes werden ignoriert.

/etc/heartbeat/authkeys

```
auth 1
1 crc
```

Ist auf beiden Servern identisch. Als Authentifizierung können verschiedene Methoden verwendet werden. crc ist Klartext und am unsichersten, wer verschlüsseln möchte kann das tun, indem folgender Eintrag in /etc/heartbeat/authkeys gemacht wird:

```
auth 1
1 sha1 Das_Passwort
```

Da das Passwort im Klartext in der Datei abegelegt wird, muss die Datei mit *chmod 600* gesichert werden.

/etc/heartbeat/haresources

```
www4  133.10.10.10     drbddisk::raid Filesystem::/dev/drbd0::/srv/raid::ext3 mysql apache2
```

Sind durch TAB/Leereintrahg getrennte Einträge

Die Datei ist ebenfalls auf beiden Servern identisch.
Im einzelnen bedeuten die Einträge:

```
www4
```

Name des Primären Webservers. Damit nach einem Reboot beider Rechner festgelegt ist, welcher Server als Primärserver straten soll, muss hier ein Rechner angegeben werden.

```
133.10.10.10
```

Ist die virtuelle IP Adresse, unter der der Webserver zu erreichen sein wird. Ist www4 nicht erreichbar, wird diese IP-Adresse auf den Ersatz-Server www5 transportiert.

```
drbddisk::raid
```

Gibt die Ressource an, die automatisch eingebunden werden soll. Hier wird festgelegt, dass mit drbd8-utils zusammengearbeitet werden soll und es sich um die definierte Ressource mit dem Namen raid handelt (siehe drbd.conf).

```
Filesystem::/dev/drbd0::/srv/raid::ext3
```

Gibt an welches Device wohin gemounted werden soll und welcher Art (hier ext3) das Filesystem ist

```
mysql apache2
```

Gibt an welche Dienste gestartet werden sollen. dazu gleich mehr.

Damit Konfigurationen wie z.B. Virtuelle Hosts beim Apache oder Änderungen an der Datenbank wirksam werden, muss ein laufender Dienst neu gestartet oder Zumindest die Konfiguration neu eingelesen werden. Übergibt heartbeat nun Dieste wie z.B. den Webservice von einem Server an den anderen, müsste auch hier ein Neustart des Servers initiiert werden.

Heartbeat kann den Start von Diensten übernehmen, was allerdings bei laufenden Diensten zu einem Fehler führen kann (z.B. Dienst bereits gestartet und es passiert nichts). Wenn heartbeat Dienste starten kann, dürfen diese Dienste also nicht bereits laufen. Der Abschluss unserer Konfiguration wird also nun sein, alle Dienste, die beim Systemstart gestartet werden, zu deaktivieren und an heartbeat zu übergeben (was wir in der haresources bereits getan haben).

Auf beiden Servern nun also noch Apache2 und Mysql aus den runlevels entfernen und wir sind fertig.

```
update-rc.d -f apache2 remove
update-rc.d -f mysql remove
```

Am besten, man stößt ein reboot auf beiden Servern an, um zu testen, ob alles geht.

Nach dem reboot sollte auf www4 der Befehl **mount** anzeigen, dass */dev/drbd0 on /srv/raid type ext3 (rw)* eingebunden wurde, **ifconfig** sollte eine Schnittstelle *eth0:0* anzeigen, auf der die IP *133.10.10.10* konfiguriert wurde. Die Dienste *apache2* und *mysql* sollten laufen (***ps aux*|*grep apache*** b.z.w ***ps aux*|*grep mysql***).

Wir testen:

www4

```
/etc/init.d/heartbeat stop
```

nach wenigen Sekunden ist /srv/raid nicht mehr eingebunden und die IP 133.10.10.10 steht nicht mehr zur Verfügung. Anders auf www5, dort sollte nun automatisch die IP-Adresse 133.10.10.10 zur Verfügung stehen und /srv/raid sollte ins System eingebunden sein. www5 sollte seine Webdienste apache2 und mysql gestartet haben. Das kann je nach Hardware ein paar Sekunden in Anspruch nehmen.

Ein Neustart von heartbeat auf www4 sollte nun www4 zum Secondary Server werden lassen und die Syncronisation des Raids starten. www5 bleibt nun solange der Primary-Server, bis er seinerseits heruntergefahren wird.

Facit:

Mit wenig Aufwand und weniger Geld lässt sich mit bestehenden (Debian) Mittel ein HA System bauen, das bisher seinen Dienst fehlerfrei tut.

Backup

Da laufende Webserver mit Forum, CMS etc ständig dynamische Daten enthalten ist es wichtig den aktiven (Primary) und nur den aktiven Server zu sichern. Dazu habe ich mir ein kleines shellscript gebastelt, dass den drbd-Status abfragt und im Fall, dass ein Server Primary ist, ein Backup auf ein NAS startet und zusätzlich die Festplatte /dev/sda1 mounted, die drbd-Ressource sichert und dann wieder aus dem System entfernt. Das Script wird über einen cronjob täglich gestartet.

Das NAS wird beim Start auf beiden Servern ins Verzeichnis /backup gemounted. Dazu habe ich mir in der Datei /etc/rc.local einen passenden mount Befehl eingetragen. /backup ist also ein NAS-Laufwerk.

Im zweiten Script (/script/save_webserver.sh) wird /dev/sdb1 nach /saving gemounted und ein copybefehl abgesetzt, ein logfile erstellt und /dev/sdb1 wieder aus dem System entfernt. Beide Scripte liegen im Verzeichnis /script.

backup.sh

```
#!/bin/bash
#
# checking drbd primary and starting backup only as primary
#
check_primary () {
  local PRIMARY=1
  local RESSTRING=
  local DRBDCMD=/sbin/drbdadm


    # Variable enthält 'Primary' oder 'Secondary'
    RESSTRING=`$DRBDCMD role all | sed -e 's/\/.*$//'`
    if [ "$RESSTRING" != "Primary" ]; then
    PRIMARY=0
        fi
        return $PRIMARY
 }


 # Falls nicht DRBD-Primary, dann Ende
 check_primary
 if [ "$?" != "0" ]; then echo "I'm primary, starting backup..."
 rsync -a /srv/raid/ /backup
 /script/save_webserver.sh
 exit 0
 fi
```

Wer kein NAS besitzt kann den rsync Befehl aus dem Script entfernen.

save_webserver.sh

```
#!/bin/bash
#
#   start backup to hard-drive
#
echo `date` " Backup started "  >> /var/log/sync.log
cd /
mount /dev/sdb1 /saving
cp /srv/raid/ /saving/ -aR
umount /saving
echo `date` " Backup completed "  >> /var/log/sync.log
echo "------------------------------------------------"  >>
/var/log/sync.log
```

save_webserver.sh hat den Schönheitsfehler, dass das Backup über 2 Server verteilt ist, je nachdem, welcher Webserver gerade aktiv ist kann es sein dass am 1.2 www4 das aktuelle Backup auf /dev/sdb1 liegen hat, am 7.2 ist das aber vielleicht www5.

Ich habe mich für den Kompromiss entschieden, da ich neben dem Backup auf die lokale Festplatte ja noch das Backup auf mein NAS habe. Ich möchte aber auf jeden Fall 2 örtlich getrennte Backups haben. Außerdem ist eine eingebundene Festplatte wesentlich schneller als eine Netzwerk Verbindung.

Was noch fehlt ist der Eintrag in die crontab von root. auf beiden Servern

crontab -e mit folgendem Inhalt:

```
30 4 * * * /script/backup.sh #Sichert den primary webserver auf
/dev/sdb1
```

startet um 4:30 täglich das Backup.

Quellen:

drbd: http://www.drbd.org/

heartbeat: http://www.linux-ha.org/wiki/Main_Page

Ubuntu ha Anleitung: http://wiki.chaosfield.org/index.php/Ubuntu_HA-Cluster_mit_DRBD_und_Heartbeat