HA OVER SAN

KRISHNA KUMAR

September 11, 2009

Abstract

This document describes making of highly available low cost SAN using FOSS. AoE target, initiator, heartbeat and drbd are main component of this SAN. This document should use in conjuction with my previous two documents. The scope of this document is limited with only step by step how to. For all the definitions, terms and terminology, please refer to my previous two documents.

1 OBJECTIVE

The objective of this experiment is to provide an end user a HA SAN so that if one node crashes, he/she will be able to use all the data and services which was running on the primary node.In this experiment there are three machines involve: 1)Primary Server, 2)Secondary Server 3)Client machine. A block device which can be a loop device(/dev/loop0), a raid device(/dev/md0) or a simple disk (/dev/hda) will be exported from primary node to client node. DRBD is responsible for providing replica of the block device from primary node to secondry node. Heartbeat takes care for network monitoring and services restart of course from primary to secondry. Lets' say suppose primary node crashes for some reason, then the corresponding replicated block device will be exported from secondary node within a few seconds and the end user which is working on the client machine, won't feel any disturbance. He will not loose any data and his work will continue.

2 Step by Step how to

2.1 System Configuration

In this experiment, three nodes are used. These are as follows with hostname and IP address :
1 CENTOS (client node, IP=10.0.0.243)
2 CENTOS1 (secondary node, IP=10.0.0.122)
3 CENTOS2 (primary node, IP=10.0.0.128)
4 virtual ip for heartbeat (shared between primary and secondary node, 10.0.0.255)

2.2 Preparing the block device with DRBD

In this experiment we have used /dev/hda8 which has to be exported from primary node to network (of course on the client node also). These are the

following steps to make drbd0 from /dev/hda8:
1 dd if=/dev/zero of=new.img bs=1M count=200
2 losetup /dev/loop0 new.img (loop device for metadata)

3 Edit /etc/drbd.conf file as follows:

```
resource testing { # name of resources
protocol C;
on CENTOS2 {
device /dev/drbd0; # Name of DRBD device
disk /dev/hda8; # Partition to use, which was created using fdisk
address 10.0.0.128:7788; # IP addres and port number used by drbd
meta-disk /dev/loop0 [0];
}
on CENTOS1{
device /dev/drbd0;
disk /dev/sda3;
address 10.0.0.122:7788;
meta-disk /dev/loop0 [0];
}
disk {
on-io-error detach;
}
net {
max-buffers 2048;
ko-count 4;
}
syncer {
rate 10M;
al-extents 257;
}
startup {
wfc-timeout 0;
degr-wfc-timeout 120; # 2 minutos. }
}
```

In this file it has mentioned that on CENTOS2 the drbd device is /dev/hda8 and on CENTOS1 it is /dev/sda3. When DRBD will start it will work as /dev/drbd0 on both the machine and it will be in sync. Copy this file on both the machine.

4 [root@CENTOS2]# scp /etc/drbd.conf root@10.0.0.122:/etc/

Run following two commands on CENTOS2 as well as on CENTOS1 secondary nodes:
5 [root@CENTOS2]# drbdadm create-md testing
6 [root@CENTOS2]# /etc/init.d/drbd restart

Make CENTOS2 as primary node by following command:
7 [root@CENTOS2]# drbdadm – –overwrite-data-of-peer primary all

Check the status of DRBD on both the machines :

8 [root@CENTOS2]# cat /proc/drbd

2.3 Configuration of heartbeat on primary and secondary node

As earlier stated heartbeat is responsible for network monitoring and services restart across primary to secondary node in case of failover. Here when we start heartbeat, it will start AoE and export the target (/dev/drbd0) over the network. If the primary node(CENTOS2) crashes /dev/drbd0 will be exported from secondary node in a few seconds so that user won't feel any disturbance and data loss. These are the following steps to configure the heartbeat:

1 Edit /etc/ha.d/ha.cf file as follows:
```
debugfile /var/log/ha-debug
logfile /var/log/ha-log
logfacility local0
keepalive 2
deadtime 30
initdead 120
bcast eth0 # Linux
auto_failback on
node CENTOS2
node CENTOS1
```

2 Edit /etc/ha.d/ha.cf file as follows:
```
CENTOS2 10.0.0.255 drbddisk::testing aoeinit.sh
```

3 Edit /etc/ha.d/authkeys file as follows:
```
auth 2
2 crc
```
The file permission of authkeys is as follows:
[root@CENTOS2]# chmod 600 /etc/ha.d/authkeys

4 Make a new file /etc/ha.d/resource.d/aoeinit.sh as follows:
```
case "$1" in
"start")
vbladed 0 3 eth0:0 /dev/drbd0
;;
"stop")
kill `pidof vblade`
;;
*)
echo "usage: `basename $0` start—stop" 1>&2
;;
esac
```

Copy all the above file from primary node to secondry node:
5 [root@CENTOS2 ha.d]# cd /etc/ha.d
6 [root@CENTOS2 ha.d]# scp ha.cf authkeys haresources root@CENTOS1:/etc/ha.d/
7 [root@CENTOS2 resource.d]# cd /etc/ha.d/resource.d

8 [root@CENTOS2 resource.d]# scp aoeinit.sh root@CENTOS1:/etc/ha.d/resource.d/

Start the heartbeat on both the nodes:
9 [root@CENTOS2 resource.d]# /etc/init.d/heartbeat start

2.4 To acess the exported block device on client side

To acess the exported block device on client side, we have to load AoE driver in existing kernel. There are two patches available for this purpose: 1)kernelpatch 2)patch_aoe. We need to apply both the patches in kernel-2.6.23 as well as in existing AoE block driver aoe6-72.

1 patch the kernel(2.6.23) by kernelpatch file whose format is as follows:

```
diff -Naur linux-2.6.23/fs/drop_caches.c one/linux-2.6.23/fs/drop_caches.c
— linux-2.6.23/fs/drop_caches.c 2007-10-10 02:01:38.000000000 +0530
+++ one/linux-2.6.23/fs/drop_caches.c 2009-09-03 14:33:02.000000000 +0530
@@ -8,6 +8,7 @@
#include <linux/writeback.h>
#include <linux/sysctl.h>
#include <linux/gfp.h>
+#include <linux/module.h> /* A global variable is a bit ugly, but it keeps
the code simple */
int sysctl_drop_caches;
@@ -44,6 +45,7 @@
}
spin_unlock(&sb_lock);
}
+EXPORT_SYMBOL(drop_pagecache);

void drop_slab(void)

@@ -53,6 +55,7 @@
nr_objects = shrink_slab(1000, GFP_KERNEL, 1000);
} while (nr_objects > 10);
}
+EXPORT_SYMBOL(drop_slab);

int drop_caches_sysctl_handler(ctl_table *table, int write,
struct file *file, void __user *buffer, size_t *length, loff_t *ppos)
```

2 [root@CENTOS linux-2.6.23]# patch -p1 <kernelpatch

3 Compile the kernel by following commands :
make menuconfig; make; make modules_install; make install

4 reboot and boot with new kernel

5 Download the latest AoE driver, patch it, compile it and install it :
6 [root@CENTOS vblade_aoe]# wget http://support.coraid.com/support/linux/aoe6-

```
72.tar.gz
7 [root@CENTOS vblade_aoe]# tar -xzvf aoe6-72.tar.gz
8 [root@CENTOS vblade_aoe]# cp patch_aoe aoe6-72
9 [root@CENTOS vblade_aoe]# cd aoe6-72
10 [root@CENTOS aoe6-72]# patch -p1 <patch_aoe
11 [root@CENTOS aoe6-72]# make; make install
12 [root@CENTOS aoe6-72]# modprobe aoe
```

If heartbeat is running on primary and secondary node then one should be able to ping 10.0.0.255(virtual ip) and he should also be able to see exported block device.

```
13 [root@CENTOS aoe6-72]# ping 10.0.0.255
PING 10.0.0.255 (10.0.0.255) 56(84) bytes of data.
64 bytes from 10.0.0.255: icmp_seq=1 ttl=64 time=0.148 ms
64 bytes from 10.0.0.255: icmp_seq=2 ttl=64 time=0.143 ms
64 bytes from 10.0.0.255: icmp_seq=3 ttl=64 time=0.147 ms
64 bytes from 10.0.0.255: icmp_seq=4 ttl=64 time=0.148 ms
64 bytes from 10.0.0.255: icmp_seq=5 ttl=64 time=0.145 ms

14 [root@CENTOS aoe6-72]# aoe-stat
e0.3 5.239GB eth0 1024 up

15 [root@CENTOS aoe6-72]# ls -l /dev/etherd/e0.3
brw-r—— 1 root disk 152, 0 2009-09-08 16:20 /dev/etherd/e0.3
```

16 Mount the exported block device and do some read write operation as follows:

```
17 [root@CENTOS aoe6-72]# mount /dev/etherd/e0.3 /mnt/

18 [root@CENTOS aoe6-72]# ls -l /mnt/
total 3128
-rw-r–r– 1 root root 0 2009-09-07 16:10 chintoocandy
-rw-r–r– 1 root root 0 2009-09-06 01:09 lo
drwx—— 2 root root 16384 2009-06-16 14:05 lost+found
-rw-r–r– 1 root root 38 2009-09-06 01:02 mm
-rw-r–r– 1 root root 0 2009-09-06 01:48 new
-rw-r–r– 1 root root 107 2009-06-16 15:02 newfile
-rw-r–r– 1 root root 47 2009-09-06 01:49 pk
-rw-r–r– 1 root root 3169720 2009-09-07 16:12 vmlinuz
```

2.5 Checking the failover across the two nodes

Now its time to see the magic. We are doing some write operation from client machine(CENTOS) to primary node(CENTOS2). While doing write operation, we will plug off the network cable. The idle case is user should not lose any data and his write operation should not disturb. These are the following steps to perform the failover phenomenon.

```
1[root@CENTOS aoe6-72]# cd /mnt/
2[root@CENTOS mnt]# dd if=/dev/zero of=new1.img bs=1M count=200
```

While copying the file block by block, plug off the network cable. Try to see the contents of /dev/drbd0 on secondary node as well as client node when dd has finished. Run following command on client node(CENTOS):

```
[root@CENTOS aoe6-72]# ls -l /mnt/
total 208132
-rw-r–r– 1 root root 0 2009-09-07 16:10 chintoocandy
-rw-r–r– 1 root root 0 2009-09-06 01:09 lo
drwx—— 2 root root 16384 2009-06-16 14:05 lost+found
-rw-r–r– 1 root root 38 2009-09-06 01:02 mm
-rw-r–r– 1 root root 0 2009-09-06 01:48 new
-rw-r–r– 1 root root 209715200 2009-09-08 17:30 new1.img
-rw-r–r– 1 root root 107 2009-06-16 15:02 newfile
-rw-r–r– 1 root root 47 2009-09-06 01:49 pk
-rw-r–r– 1 root root 3169720 2009-09-07 16:12 vmlinuz
```

Run following command on secondary node:

```
[root@CENTOS1 ]# ifconfig eth0:0
eth0:0 Link encap:Ethernet HWaddr 00:14:85:7C:89:0E
inet addr:10.0.0.255 Bcast:10.0.3.255 Mask:255.255.252.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
Interrupt:16 Base address:0x4000
[root@CENTOS1 ]# mount /dev/drbd0 /mnt/
[root@CENTOS1 ]# ls -l /mnt/
total 208132
-rw-r–r– 1 root root 0 Sep 7 16:10 chintoocandy
-rw-r–r– 1 root root 0 Sep 6 01:09 lo
drwx—— 2 root root 16384 Jun 16 14:05 lost+found
-rw-r–r– 1 root root 38 Sep 6 01:02 mm
-rw-r–r– 1 root root 0 Sep 6 01:48 new
-rw-r–r– 1 root root 209715200 Sep 8 17:30 new1.img
-rw-r–r– 1 root root 107 Jun 16 15:02 newfile
-rw-r–r– 1 root root 47 Sep 6 01:49 pk
-rw-r–r– 1 root root 3169720 Sep 7 16:12 vmlinuz
```

So, what we see that even if primary node crashes, our data will be saved on secondary node. Client doesn't feel any disturabnce while working on exported block device. Virtual ip has also shifted from primary node to secondary node.