DRBD Setup for Production

Scenario:
DB2-Primary
DB1-Secondary
Operating system: Red Hat Enterprise Linux AS release 4
Download the latest tar from http://oss.linbit.com/drbd/8.2/drbd-8.2.5.tar.gz

untar the package drbd-8.2.5.tar.gz

#**tar –xvzf drbd-8.2.5.tar.gz**
#**cd drbd-8.2.5**
#make rpm
This would build the RPM's specific to the architecture.

1) Follow the procedure as below on DB1 and DB2

Install the RPM's

1) drbd-8.2.5-3
2)drbd-debuginfo-8.2.5-3
3)drbd-km-2.6.9_42.ELsmp-8.2.5-3

Insert the drbd module into kernel (optional not required in most cases)
#modprobe drbd (Optional)

Configuration file to be updated /etc/drbd.conf:
Note:Configuration file drbd.conf should be identical on both DB1/DB2

```
# Our MySQL share
resource db {
  protocol C;
#  incon-degr-cmd "echo '!DRBD! pri on incon-degr' | wall ; sleep 60 ; halt -f";
#  startup { wfc-timeout 0; degr-wfc-timeout     120; }
  disk { on-io-error detach; } # or panic, ...
  syncer {
     rate 100M;
     verify-alg md5;
            }
   net {
    allow-two-primaries;
  }
```

```
  on db1.habeas.com {
    device /dev/drbd1;
    disk /dev/sdb1;
    address 192.168.0.1:7789;
    meta-disk internal;
  }
  on db2.habeas.com {
    device /dev/drbd1;
    disk /dev/sdb1;
     address 192.168.0.2:7789;
     meta-disk internal;
      }
}
```

On Both DB1/DB2 Execute

dd if=/dev/zero of=/dev/sdb1 bs=1M count=128 (This would enable meta-disk internal)
drbdadm create-md db

/etc/init.d/drbd start

```
#cat /proc/drbd or
```
#service drbd status
which show the following details

- cs — connection state
- st — node state (local/remote)
- ld — local data consistency
- ds — data consistency
- ns — network send
- nr — network receive
- dw — disk write
- dr — disk read
- pe — pending (waiting for ack)
- ua — unack'd (still need to send ack)
- al — access log write count

Initializing DRBD

```
On DB2:

Make DB2 as primary server.

db2#drbdadm -- --overwrite-data-of-peer primary db
```

On DB1

```
[root@db1 ~]#mkfs.ext3 -L mysql /dev/drbd1
[root@db1 ~]#drbdadm adjust db
[root@db1 ~]#drbdadm secondary db
```

On DB2:

```
[root@db2 ~]# drbdadm adjust db
[root@db2 ~]# drbdadm primary db
[root@db2 ~]# mount /dev/drbd1 /var/lib/mysql
```

Configuring MySQL to use DRBD

Edit my.cnf so datadir=/var/lib/mysql. After that, run mysql_install_db. Check /var/lib/mysql to see if the mysql database directory has been created, and all the files are there.
Start mysql:

Make sure you see that mysqld is using databases from /var/lib/mysql in db1 or /var/lib/mysql in db2.

In case of failure of primary DB server

On Secondary server:
#drbdadm primary all
#mount /dev/drbd1 /var/lib/mysql
#start the MySQL Process.

On Primary Server:
#stop the MySQL Process
Unmount the DRBD block Device
#umount /dev/drbd1
#drbdadm secondary all

Now the sync would happen from secondary to primary DB server.

Once the problem with Primary DB Server is fixed

On Primary Server:
#drbdadm primary all
#mount /dev/drbd1 /var/lib/mysql
#start the MySQL Process

On Secondary server:
#stop the MySQL Process
#umount /dev/drbd1
#drbdadm secondary all

At this point the sync is happening from primary to secondary
Split Brain error:

At times when both the Primary and secondary server status shows as standalone and refusing to get connected/sync and also you see “Split-Brain detected” in “dmesg”

drbdadm -- --discard-my-data connect all #(on node with "bad" data)
drbdadm connect all #(on node with "good" data)

This should make both the DB Servers to connect and start syncing the data.