September 2003

DB2 Information Management Software

Open Source Linux High Availability for IBM® DB2® Universal Database™

Implementation Guide

By
Baris Naciterhan
IBM Software Group

Contents

1. INTRODUCTION

1.1 Objective

This white paper describes the setup and implementation of Linux High Availability solution for IBM® DB2® Universal Database™. In this step-by-step guide, you will find all the required information to set up and run the High Available DB2 UDB Database.

1.2 Overview

The most important asset a company has is the data stored in their database. There are numerous circumstances that can take a system off-line, ranging from planned downtime for maintenance to catastrophic failure. To prevent the company from having any type of failure which can cause data loss, redundancy is used.
A system that has a hardware and software redundancy is called a High Available system. Linux HA creates a secure environment for DB2 UDB by supporting software and hardware redundancy.
Only a system consisting of 2 machines is supported by Linux High Availability (HA) right now: a primary (main) node and a secondary node. The goal of Linux HA is to configure the system so that the workload can be taken over by the secondary node when a failure occurs on the primary node. To achieve this, the components of Linux High Availability should be installed and configured. These components are:

Drbd: The component which performs the data synchronization between the nodes in the cluster.

Heartbeat: The component which prepares the environment for the constituent nodes of the cluster to communicate with each other for detecting node or daemon failures and for reconfiguring the system appropriately.

Contents

2. LINUX HA SETUP

The Linux High Availability setup process has 2 parts: Hardware and Software setup. The performance of the Linux High Availability cluster depends on both hardware and software properties of the whole system.

2.1 Basic Requirements

- ✓ 2 servers connected to the local network.
- ✓ A cross-over cable
- ✓ Linux Operating System on both machines
- ✓ Null Modem Cable (optional for additional redundancy)
- ✓ DB2 UDB Enterprise version 8.0 (or higher) for both machines

Note that in order to configure and run Linux High Availability cluster you should log in as root.

2.2 Hardware Setup

Each node in the cluster must have two network interface cards (NIC). One network card on each node of the cluster should be connected to the external network. The other network card should be connected to the network card on the other node via a cross-over cable. For the redundancy of the communication link, a null modem cable can be used between the nodes as a serial connection. Being identical is not mandatory for the nodes in the cluster, however, the shared partition must be exactly the same on each node. While creating the shared partition, do not mount it since Drbd will mount the shared partition. The actual sizes of the hard disks on the nodes can be different but as the shared partition is for mirroring data between the nodes in the cluster, the size and type of the shared partition must be the same on each node. Due to bandwidth issues, the size of the shared partition should be limited to what is necessary to hold the shared DB2 UDB data only. Also note that larger shared partitions require more time for full synchronization. Figure 1 shows the Linux High Availability hardware setup.

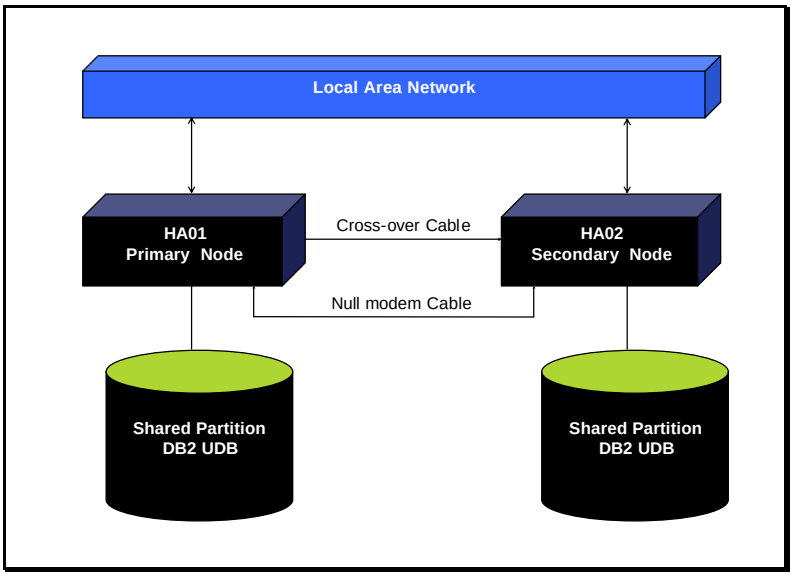


Figure 1: *DB2 UDB Linux High Availability hardware schema*

2.3 Software Setup

- **SuSE:**

SuSE fully supports Linux High Availability and the SuSE Linux installation CDs contain the full Linux High Availability package. Installing the latest Linux SuSE service pack, however, is a requirement for Linux High Availability. Heartbeat and Drbd are standard packages but updating to the latest stable versions is highly recommended. By updating to the latest version of the service pack, you will get current levels of these components with significant enhancements. If your SuSE version is old, it may not include the HA package. In that case please refer to the websites listed below in order to download the most current HA package.

- **RedHat and other Linux distributions:**

Linux High Availability is supported on RedHat but the RedHat Linux installation cds do not contain the Linux High Availability package right now. Therefore, HA components (Drbd and Heartbeat) should be installed depending on the RedHat version using the links below. HA files are also available for other supported Linux distributions on these sites.

Where to download HA components:

Heartbeat: http://www.linux-ha.org/download/
http://www.ultramonkey.org/download/heartbeat/
Drbd: http://www.complang.tuwien.ac.at/reisner/drbd/download/

There are four main files that need to be configured to make HA run:

drbd.conf: This is the Drbd configuration file and it includes information about the nodes and the shared partition. This file must be identical on both nodes.

authkeys: This is one of the Heartbeat configuration files and it is for the encryption of the Heartbeat packets between the nodes. For controlling the process, this file includes the key and because of that only root has permissions on this file. For authkeys you must set permissions only for root, otherwise HA will not run. This file must be identical on both nodes.

ha.cf: This file contains various specific parameters for Heartbeat. Using these parameters communication link between the nodes can be modified on behalf of the system needs. This file will be different between the primary and secondary machines due to the IP addresses.

haresources: This file contains information about the primary node and the resources that move from node to node in case of a failure. This file must be identical on both nodes.

3. TEST ENVIRONMENT

As primary and secondary nodes are identical, the following configuration applies to both. Note that by using more powerful IBM hardware equipments, you can reduce the failover time.

- Machine : IBM Xseries 330
- *CPU* : Intel(R) Pentium(R) III 1133MHz
- *Memory* : 512 MB
- *NIC* : Intel Ethernet Pro 100
- *Hard Disk* : SCSI 18GB, SCSI 36GB

3.1 General Characteristics of the Linux HA Cluster

Each node has two NICs.The first NIC (eth0) on each node is connected to the local network. The second NIC (eth1) on each node is connected to the second NIC (eth1) on the other node. As a result five IP addresses should be configured for the cluster.
Four IP addresses for the two NICs on each node and one for representing the whole cluster. But it is **crucial** that the IP address for the whole cluster should be configured only in the 'haresources' file.

3.2 IP Setup of the Test Environment

·*Primary Node (**ha01**)*
Eth0 (connected to local network) : 9.26.166.188
Eth1 (connected to eth1 on the secondary) : 172.16.0.1

·*Secondary Node (**ha02**)*
Eth0 (connected to local network) : 9.26.166.182
Eth1 (connected to eth1 on the secondary) : 172.16.0.3

·*The **whole** cluster* : 9.26.167.95

Figure 2 shows the IP schema of the Linux High Availability test cluster:

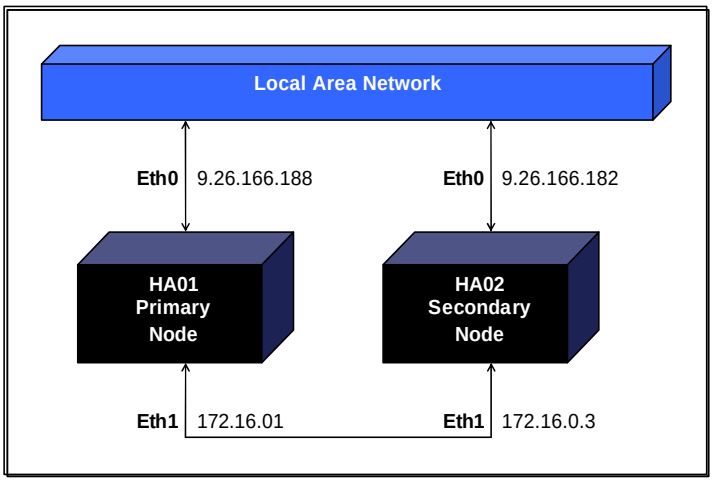


Figure 2: *IP Schema of the Test Cluster*

4. CONFIGURING LINUX HA CLUSTER

4.1 General Configuration

Before configuring Linux HA, the standing cluster environment should be generally configured:

Step 1: Checking the Heartbeat folder '/etc/ha.d/'

Heartbeat requires certain files to be stored in **/etc/ha.d/**
To check for the files type the following:

```
# ls –l /etc/ha.d/
```

If you do not see all of the Heartbeat files (authkeys, ha.cf, haresources) in /etc/ha.d/, copy them into this folder.
The location of these files depends on the operating system:

- **SuSE:**

On SuSE, the files are located in '/usr/share/doc/packages/heartbeat/' right now. To copy the files into the required folder, type the following commands:

```
# cp /usr/share/doc/packages/heartbeat/authkeys
/etc/ha.d/

# cp /usr/share/doc/packages/heartbeat/ha.cf
/etc/ha.d/

# cp /usr/share/doc/packages/heartbeat/haresources
/etc/ha.d/
```

- **RedHat and other Linux distributions:**

On RedHat, the files are located in '/usr/share/doc/heartbeat-1.0.3/' right now. To copy the files into the required folder, type the following commands:

```
# cp /usr/share/doc/heartbeat-1.0.3/authkeys
/etc/ha.d/

# cp /usr/share/doc/heartbeat-1.0.3/ha.cf
/etc/ha.d/

# cp /usr/share/doc/heartbeat-1.0.3/haresources
/etc/ha.d/
```

Depending on the O.S. version, if you don't see the files in the folders written above, use *find* command in order to search for the needed files.

To test: You should see the copied files (authkeys, ha.cf, haresources) when you type:

```
# ls –l   /etc/ha.d/
```

Step 2: Link to /etc/init.d/drbd

You should see the datadisk symbolic link pointing to '/etc/init.d/drbd' when you type the following command:

```
# ls  -al  /etc/ha.d/resource.d/
```

If you do not see the symbolic link for mounting the shared device, a symbolic link should be created. Simply type the following commands:

```
# cd /etc/ha.d/resource.d
```

```
# ln –s /etc/init.d/drbd datadisk
```

4.2 Drbd Configuration

The Drbd component can be configured with the *drbdsetup* command. But the easiest way of configuring Drbd is to modify the ***drbd.conf*** file. This file must be the same on both nodes.

```
# vi /etc/drbd.conf
```

The contents of the file should be similar to the listing below:

```
resource drbd0 {

  protocol = C
  fsckcmd  = fsck.ext3 -p -y
#  inittimeout=60
#  skip-wait

  disk {
    do-panic
    disk-size = 2160711
```

```
}
 # you can reconfigure the sync rate at runtime using
 # drbdsetup on the primary.

  net {
#    skip-sync
#    sync-min
#    sync-rate         # synonym for sync-max
    sync-nice  = -18  # if synchronization is high priority for you
    sync-max    = 100M # if you don't care about network saturation
#    sync-max    = 250
    tl-size     = 5000
    timeout     = 60
    connect-int = 10
    ping-int    = 10
}
 on ha01 {
    device  = /dev/nb0
    disk    = /dev/sdb1
    address = 172.16.0.1
    port    = 7788
}
 on ha02 {
    device   = /dev/nb0
    disk     = /dev/sdb1
    address  = 172.16.0.3
    port     = 7788
}
}
```

Explanation of *drbd.conf:*

If you need only one device (like Heartbeat), you will need only one main configuration block (drbd0). Otherwise the configuration blocks must be named as drbd0, drbd1, …and default port numbers are 7788 (drbd0) , 7789 (drbd1), …

At first protocol *type* should be selected. Valid protocol types are:

A: A write operation is complete as soon as the data is written to disk and sent to the network

B: A write operation is complete as soon as a reception acknowledgement arrives

C: A write operation is complete as soon as a write acknowledgement arrives.

Selecting the *disk size* is the second important point. The disk-size must be the same as the shared partition size and should be identical on both nodes. Set the disk-size depending on your DB2 database needs. For instance on the testing HA cluster, partition type (fsckcmd field) is ext3, on /dev/sdb1(disk field) with the disk size (disk-size field) 2160711(2GB). Additionally, the *do-panic* option is set so that Drbd triggers a kernel panic in case of an IO error on the lower_device.
The last two blocks contain the specific information about nodes. Other options that may be appended to 'drbd.conf' are:

Timeout: If the communication is blocked for the given value * 1/10 seconds, Drbd falls back into unconnected operation. The default value is 30.

Sync-rate: The synchronization rate (KB per sec). The default value is 250

Skip-sync: Instructs Drbd not to do synchronization.

Tl-size: Sets the size of the transfer log (TL). The TL is used for dependency analysis. For long latency high bandwidth links, it might be necessary to set the size greater than 256. You will find error messages in the system log if the TL is too small. The default value is 256.

Connect-int: This is the total number of seconds required for Drbd to search for a connection. The default value is 10.

Ping-int: By not allowing this feature, secondary node can not distinguish a broken link from an idle link.
If the link is idle longer than a specified time in seconds, Drbd will send a no-op packet over the link to inform the node. The default value is 10.

4.3 Heartbeat Configuration

In order for Heartbeat to run, three files should be configured; 'authkeys', 'ha.cf' and 'haresources'.

- **authkeys:**

This is the authentication file for synchronization and it controls the encryption of the Heartbeat packets. This file must be identical on both nodes. The format of this file is:

auth <number> <number> <method> "key string"

Available methods are:

Sha1: Full secure authentication regardless of CPU resources.

Md5: Secure authentication (less secure than sha1) with minimized CPU resources.

Crc : To be used only on physically secure networks. It adds no security except from packet corruption, thus no need to use "key string".

Type the following command to check the content of the file:

```
# vi /etc/ha.d/authkeys
```

The contents of the file should be similar to the listing below:

```
#
#       Authentication file.  Must be mode 600
#
#
#       Must have exactly one auth directive at the front.
#       auth    send authentication using this method-id
#
#       Then, list the method and key that go with that method-id
#
```

```
#       Available methods: crc sha1, md5.  Crc doesn'tneed/want a key.
#
#       You normally only have one authentication method-id listed in this
file
#
#       Put more than one to make a smooth transition when changing auth
#       methods and/or keys.
#
#
#
#       crc adds no security, except from packet corruption.
#               Use only on physically secure networks.
#
#auth 1
#1 crc
#2 sha1 HI!
auth 1
1 md5 Deneme
```

After modifying the file, type the following command to be sure that its permissions are secure.

```
# chmod 600 /etc/ha.d/authkeys
```

- **ha.cf:**

This file contains the configurable Heartbeat parameters for the network communication between the nodes. Due to their IP addresses, this file should be different on both nodes.

Type the following command:

```
# vi /etc/ha.d/ha.cf
```

The contents of the file should be similar to the output below:

```
#       There are lots of options in this file.  All you must have is a set
#       of nodes listed {"node ...}
#       and one of {serial, bcast, mcast, or ucast}
#
node ha01
node ha02
keepalive 2

#baud 19200
#udpport 694
serial /dev/ttyS0
bcast eth1
deadtime 10
nice_failback on
```

You can append many options to this file depending on the configuration of your network. Basically you need to write the name of the nodes and the private IP address of each node in this file in order to run Heartbeat. The configurable parameters you can use in ha.cf are:

node ha01, node ha02: These are the names of the nodes. You can learn yours by typing:

```
# uname  -n
```

The first node listed in the file must be the name of the primary node.

bcast eth1: The NIC eth1 which is used for the communication with the other node.

deadtime 10: This is the time taken, in seconds, to declare a node is dead.

nice_failback on: In case of a failure on the primary node**,** secondary takes over and when the primary node becomes active again, it gets everything back from the secondary node. This option prevents the primary node from re-acquiring cluster resources after a failover.

serial /dev/ttyS0: This is the device to be used via a null modem cable for serial Heartbeat connection.

keepalive 2: This is the time in seconds between heartbeats.

- **Haresources:**

This file specifies the primary node and the services of the cluster that will be moved between the nodes in case of a failover. Haresources also includes the IP address of the whole cluster which should not be configured outside of this file. Haresources file must be the same on both nodes. To configure haresources, type the following command and modify the file according to your needs:

```
# vi /etc/ha.d/haresources
```

The contents of the file should be similar to the listing below:

```
#
#       The string you put in for nodename must match the uname -n name
#        of your machine.  Depending on how you have it administered, it could
#       be a short name or a FQDN.
#
#-------------------------------------------------------------------
#
#       Simple case: One service address, default subnet and netmask
#               No servers that go up and down with the IP address
#
#just.linux-ha.org      135.9.216.110
#
#-------------------------------------------------------------------
#
#       Assuming the administrative addresses are on the same subnet...
#       A little more complex case: One service address, default subnet
#       and netmask, and you want to start and stop http when you get
#       the IP address...
```

```
#
#just.linux-ha.org      135.9.216.110 http
#-------------------------------------------------------------------
#
#           A little more complex case: Three service addresses, default subnet
#       and netmask, and you want to start and stop http when you get
#       the IP address...
#
#just.linux-ha.org      135.9.216.110 135.9.215.111 135.9.216.112 httpd
#-------------------------------------------------------------------
#
#       One service address, with the subnet, interface and bcast addr
#       explicitly defined.
#
#just.linux-ha.org      135.9.216.3/28/eth0/135.9.216.12 httpd
#
#-------------------------------------------------------------------
#
#       An example where a shared filesystem is to be used.
#       Note that multiple aguments are passed to this script using
#       the delimiter '::' to separate each argument.
#
#node1  10.0.0.170 Filesystem::/dev/sda1::/data1::ext2


ha01 9.26.167.95 datadisk::drbd0
```

***Definition of haresources*:**

ha01 9.26.167.95 datadisk::drbd0

ha01: The name of the primary node. Note that haresources file must include only the name of the primary node. I could also write like: ha01.torolab.ibm.com.

9.26.167.95: The virtual IP address of the cluster. **Do not write** this address other than in the haresources file.

datadisk::drbd0: This starts datadisk service using drbd0.

As an *alternative*, you can also include the network properties in the haresources file. For instance, written below represents the haresources file including the network properties of the HA cluster:

ha01 *9.26.167.95/26/eth0/9.26.167.255* datadisk::drbd0

26: Subnet (based on netmask).

eth0: The network device on which the virtual IP address will be created.

9.26.167.255: The broadcast address of the cluster.

4.4 /etc/fstab Configuration

This file includes information about the mounted devices.
Because of that before modifying the *fstab* file, you must be sure that the the shared partition is free to be mounted. Note that if there is already a row which contains shared partition in the *fstab* file, you must modify that row as written below. You should modify the **fstab** file on both nodes. Simply type the following command:

```
# vi /etc/fstab
```

The contents of the file should be similar to the output below:

```
devpts /dev/pts devpts    mode=0620,gid=5   0 0
proc  /proc  proc       defaults          0 0
usbdevfs /proc/bus/usb usbdevfs noauto              0 0
/dev/cdrom  /media/cdrom auto      ro,noauto,user,exec   0 0
/dev/fd0          /media/floppy       auto       noauto,user,sync      0 0
/dev/nb0        /db2ha              ext3      rw,noauto             0 0
```

Except for the last line, all are the usual system mount information.

/dev/nb0 /db2ha ext3 rw,noauto 0 0: Mount the device /dev/nb0 to the physical mount point /db2ha with the filesystem type ext3 having the read/write permissions. *noauto 0 0* function prevents the system from mounting the device automatically as only datadisk service mounts the device.

4.5 Preparing the shared partition

After creating the shared partition and configuring Heartbeat, the file system should be prepared for Linux High Availability. Depending on your system, type the following commands:

```
# mke2fs –j /dev/sdb1

# mkdir /db2ha
```

The first command creates the journaling file system for the shared partition and the second command will create the mount point.

5. STARTING LINUX HA

Drbd must be started before Heartbeat, and both should first be started on primary server and then on the secondary server.
Depending on the operating system, the locations of the files are:

- ***SuSE***

Drbd: /etc/init.d/drbd
Heartbeat: /etc/init.d/heartbeat

- ***RedHat and other Linux distributions***

Drbd: /etc/rc.d/init.d/drbd
Heartbeat: /etc/rc.d/init.d/heartbeat

Depending on the version, the location of Drbd and Heartbeat files can be different. As a result, the best way to check for both is to use the find command. Type the command below to find the files and start them from their locations on both nodes in the same order.

```
# find / -name drbd

# find / -name heartbeat
```

5.1 Starting Drbd

Depending on the operating system, first on primary node and then on secondary node type the following commands:

- ***SuSE***

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

- ***RedHat and other Linux distributions***

```
# /etc/rc.d/init.d/drbd start
```

*On the **primary node** (ha01):*

```
ha01:~ # /etc/init.d/drbd start
Setting up 'drbd0'...[ OK ]
waiting for connection: drbd0
Do you want to abort waiting for other server and make this one primary?
```

Don't respond to 'abort waiting' prompt after starting Drbd on the primary node, otherwise synchronization between the nodes cannot succeed.

*On the **secondary node** (ha02):*

```
ha02:~ # /etc/init.d/drbd start
Setting up 'drbd0'...[ OK ]
waiting for connection: drbd0
Waiting until 'drbd0' is up to date (using SyncingAll)
[1116] wait_sync 'drbd0'
All resources connected.
```

Note that synchronization time depends on the size of the shared partition.

5.2 Starting Heartbeat

Depending on the operating system, first on primary node and then on secondary node type the following commands:

- ***SuSE***

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

- ***RedHat and other Linux distributions***

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

6. MONITORING LINUX HA

Although Drbd has a specific file for monitoring, Heartbeat has none. However, the status of Heartbeat can be monitored from the general log file of the Linux operating system.

6.1 Monitoring Drbd

Drbd has a specific file which includes the information of its status. Type the following command to see the content of the file:

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

If you type this command on the primary server before the completion of the synchronization process, you will see a response similar to the following:

```
version: 0.6.3 (api:61/proto:62)
0: cs:SyncingAll st:Primary/Secondary ns:1212888 nr:0 dw:0 dr:1212900 pe:154 ua:0
        [===========>.....] sync'ed: 56.2% (925/2110)M
        finish: 1:22min speed: 11,496 (11,443) K/sec
```

If you check the file after the completion of the synchronization process, you should see similar response to the one below:

*On the **primary** node:*

```
ha01:~ # cat /proc/drbd
version: 0.6.3 (api:61/proto:62)
0: cs:Connected st:Primary/Secondary ns:2160708 nr:0 dw:0 dr:2160708 pe:34 ua:0
```

*On the **secondary** node:*

```
ha02:~ # cat /proc/drbd
version: 0.6.3 (api:61/proto:62)
0: cs:Connected st:Secondary/Primary ns:0 nr:2160708 dw:2160708 dr:0 pe:0 ua:0
```

Also by using command *lsmod,* you can check whether Drbd kernel modules are loaded or not:

```
ha01:~ # lsmod
Module                  Size  Used by    Not tainted
drbd                   37640   2
videodev                5600   0  (autoclean)
ide-cd                 28388   0  (autoclean)
isa-pnp                29664   0  (unused)
ipv6                  209692  -1  (autoclean)
st                     26924   0  (autoclean) (unused)
sr_mod                 13432   0  (autoclean) (unused)
cdrom                  26400   0  (autoclean) [ide-cd sr_mod]
sg                     27328   0  (autoclean)
joydev                  5600   0  (unused)
evdev                   4352   0  (unused)
input                   3168   0  [joydev evdev]
usb-ohci               18184   0  (unused)
```

6.2 Monitoring Heartbeat

After starting Heartbeat, wait at least 10 seconds before checking the log file. As Heartbeat starts different services in order to run HA, it takes time to finish the whole process. Type the following command in order to monitor Heartbeat:

```
# cat /var/log/messages | grep heartbeat
```

This command will check the general Linux log file for the lines including Heartbeat.

The contents of the file should be similar to the listing below:

```
ha01 heartbeat: info: Running /etc/ha.d/resource.d/datadisk drbd0 start
ha01 heartbeat: debug: Starting /etc/ha.d/resource.d/datadisk drbd0 start
ha01 heartbeat: debug: /etc/ha.d/resource.d/datadisk drbd0 start done. RC=0
ha01 heartbeat: /usr/lib/heartbeat/send_arp eth0 9.26.167.95 00096B588B89 9.26.167.95 ffffffffffff
ha01 heartbeat: /usr/lib/heartbeat/send_arp eth0 9.26.167.95 00096B588B89 9.26.167.95 ffffffffffff
ha01 heartbeat[1023]: info: Local Resource acquisition completed. (none)
ha01 heartbeat[1023]: info: local resource transition completed.
```

6.3 Monitoring the filesystem

After starting Drbd and Heartbeat, the physical mount point (/db2ha) for the shared partition should have been mounted by the network block device (/dev/nb0) on the primary node. Mounting should have been completed only by the active node. When HA is being activated, before a failover, you should not see your shared partition mounted on the secondary node as the active node is the primary node.

*On the **primary** node:*

```
ha01:~ # mount
/dev/sda1 on / type reiserfs (rw)
proc on /proc type proc (rw)
devpts on /dev/pts type devpts (rw,mode=0620,gid=5)
/dev/sda3 on /boot type ext3 (rw)
shmfs on /dev/shm type shm (rw)
usbdevfs on /proc/bus/usb type usbdevfs (rw)
/dev/nb0 on /db2ha type ext3 (rw)
```

*On the **secondary** node:*

```
proc on /proc type proc (rw)
devpts on /dev/pts type devpts (rw,mode=0620,gid=5)
/dev/sda3 on /boot type ext2 (rw)
```

```
shmfs on /dev/shm type shm (rw)
usbdevfs on /proc/bus/usb type usbdevfs (rw)
```

If you see your shared partition mounted on both nodes, check the file /etc/fstab and erase the rows containing your shared partition except the one appended for datadisk script (/dev/nb0 /db2ha ext3 noauto 0 0). Mounting should have been completed only by the active node. Only the active side will have the filesystem mounted.

The physical mount point on each node can be checked using the *ls* command:

*On **primary** node:*

```
ha01:~ # ls /db2ha
. ..  lost+found
```

*On **secondary** node:*

```
ha02:~ # ls /db2ha
. ..
```

As you see, right now the folders are empty on both nodes. On the next section, a simple file will be created in order to test the HA cluster.

7. SIMPLE FAILOVER TEST

If there are no errors in the log files after starting Drbd and Heartbeat, test the HA cluster. Otherwise, you should check both hardware configurations (server-cable installation) and the software configurations (Drbd-Heartbeat) of the HA cluster.
Assuming that Drbd and Heartbeat were started in the same order on both nodes (Drbd first) without an error, run the following commands to test the Linux HA Cluster:

On the primary node:

```
# cd  /db2ha

# ls  -l  > deneme
```

```
ha01:/db2ha # ls
```

```
. .. deneme  lost+found
```

We created our test file on the primary node. The next process is causing a virtual failure to test the cluster. The easiest way to test the failover is simply to reboot the primary node. After rebooting the primary node, in a few seconds the files in the mounted partition and services of the node written in the haresource file should be transferred into the secondary server:

```
# shutdown  –fr  now
```

On the secondary node:

Wait a few seconds and check the log file on the secondary node, using the following command to find events related to Heartbeat:

```
# cat /var/log/messages | grep heartbeat
```

The contents of the file should be similar to the output below:

```
ha02 heartbeat: info: /usr/lib/heartbeat/mach_down: nice_failback: acquiring foreign resources
ha02 heartbeat[913]: info: mach_down takeover complete.
ha02 heartbeat: info: mach_down takeover complete for node ha01.
```

Check whether if the partition is mounted:

```
ha02:/ # mount | grep db2ha
/dev/nb0 on /db2ha type ext3 (rw)
```

And finally if everything is running as anticipated, check the folder **/db2ha**. You should see ‘deneme’; the file we created on the primary server.

```
ha02:/ # ls /db2ha
. .. deneme  lost+found
```

Adding Drbd and Heartbeat to system startup

If the failover test runs successfully, add Drbd and Heartbeat to the system startup as starting them manually takes longer. Type the following commands in the order shown:

```
# chkconfig --add heartbeat
# chkconfig --add  drbd
```

8. CONFIGURING DB2 UDB FOR LINUX HA

Configuring DB2 UDB for Linux High Availability means:

→ Installing DB2

→ Adding DB2 service to the Heartbeat haresources file

→ Creating & Configuring DB2 database for HA

8.1 Installing DB2 UDB

To Install DB2, use the standard installation process. No extra tool or component is needed for Linux High Availability. While installing DB2, the only changes needed are to certain DB2 settings on each node. DB2 files on the nodes will be associated with the numerical Group ID (gid) and the User ID (uid) fields. It is crucial that these values match on both machines. Otherwise DB2 Linux High Availability will encounter permission errors.

For example, imagine that on the primary node, the user db2inst1 who has a uid of 101and gid of 200, writes the database to the shared partition. Even the user with the same name on the secondary node will not be able to read files written by the primary node without having the same numerical uid and gid. As an illustration, written below are the DB2 group IDs (gid) and user IDs (uid) on both nodes of the HA cluster:

Group ID	Group Name	Group Members
101	Dasadm1	dasusr1,db2inst1
102	Db2fgrp1	db2fenc1
104	Db2grp1	db2inst1

User ID	User Name
103	Dasusr1
105	Db2inst1
106	Db2fenc1

After completing the DB2 installation, the '/etc/inittab' file should be modified on both nodes to run DB2 HA correctly. Comment out the line which starts db2 on system startup as follows:

```
# fmc:2345:respawn:/opt/IBM/db2/V8.1/bin/db2fmcd #DB2 Fault Monitor Coordinator
```

8.2 Adding the DB2 service to the Heartbeat haresources file

After the DB2 installation is completed, the next step is modifying haresources file for DB2 UDB service. Type the following commands: The string; **db2::db2inst1** which starts db2 service with the instance named as db2inst1 to launch the instance on the mirrored partition should be appended to the Heartbeat haresources file with the location; /etc/ha.d/haresources on both nodes. After adding the string to the file, the content of the file should be similar to the one below:

```
ha01  9.26.167.95 datadisk::drbd0 db2::db2inst1
```

After modifying the files on both nodes, the machines should be rebooted in order to use the DB2 service. Follow these steps in the order listed:

On the **primary** node:

```
# shutdown –fr now
```

After the machine becomes active again, log on as root.

On the **secondary** node:

```
# shutdown –fr now
```

After the machine becomes active again, logon as root.

8.3 Creating & Configuring DB2 database for HA

The next step is to create the DB2 database. Type the following commands in the same order to create and configure the database.

*On the **primary** node:*

1- In order to work in the DB2 database environment, log on as the db2 instance user:

```
ha01:~# su db2inst1
```

2- Type the following commands with the same order to activate DB2:

```
db2inst1@ha01:/> db2start
```

You should get one of the following two messages:

```
0   0   SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
```

```
0   0   SQL1026N  The database manager is already active.
SQL1026N  The database manager is already active.
```

Any other message probably indicates that your instance is not correctly configured. In that case, check the log files for errors and reconfigure your DB2 instance.

```
db2inst1@ha01:/> db2jstrt
```

```
Licensed Materials -- Property of IBM
(c) Copyright International Business Machines Corporation, 1996, 2000.
   All Rights Reserved.
```

3- Create the DB2 database to be highly available:

```
db2inst1@ha01:/> db2 create database db2ha on /db2ha
```

You should see the message below:

```
DB20000I  The CREATE DATABASE command completed successfully.
```

At this point database is created on primary server. To make the database accessible to the secondary node, type:

```
db2inst1@ha01:/> exit
```

```
ha01:~# shutdown –fr now
```

*On the **secondary** node:*

1- Logon as the db2 instance user.:

```
ha02:~# su db2inst1
```

2- Type the following command in order to catalog the database:

```
db2inst1@ha02:/ > db2 catalog database db2ha on /db2ha
```

You should see the message below:

```
DB20000I  The CATALOG DATABASE command completed successfully.
```

Type the following commands to finish the process:

```
db2inst1@ha02:/ > exit
```

```
ha02:~ # shutdown –fr now
```

9. DB2 UDB LINUX HA FAILOVER TEST

We are ready to test the DB2 HA system. The test process consists of:

- Creating a table on the primary node in the database named as *db2ha* which was created above.

- Causing the primary to fail so that the secondary can access to the shared file system and the db2 database.

- Checking for the table in db2ha on the secondary node and inserting some values to the table.

- Causing the secondary to fail so that the primary node can get back the db2 database.

Step 1: We will work at the primary node. After creating the table named *failover_test* at db2ha, we will cause a failover on the primary node by rebooting the machine.

- Login to the primary node as db2 instance user and check the shared folder /db2ha for the database:

```
ha01:~ # su db2inst1
```

```
db2inst1@ha01:/root> ls  /db2ha
```

```
db2  db2inst1  lost+found
```

- Connect to the database db2ha:

```
db2inst1@ha01:/root> db2 connect to db2ha
```

```
 Database Connection Information

 Database server        = DB2/LINUX 8.1.2
 SQL authorization ID   = DB2INST1
 Local database alias   = DB2HA
```

- Check the status of the database:

```
db2inst1@ha01:/root> db2 list active databases
```

```
                 Active Databases

 Database name                   = DB2HA
 Applications connected currently= 1
 Database path                   =
/db2ha/db2inst1/NODE0000/SQL00001/
```

- Enter Db2 command line processor and create the table ‘failover_test’ with the fields; month and year:

```
db2inst1@ha01:/root> db2
```

```
(c) Copyright IBM Corporation 1993,2002
Command Line Processor for DB2 SDK 8.1.2

You can issue database manager commands and SQL statements from
the command prompt. For example:
   db2 => connect to sample
   db2 => bind sample.bnd

For general help, type: ?.
For command help, type: ? command, where command can be
the first few keywords of a database manager command. For example:
 ? CATALOG DATABASE for help on the CATALOG DATABASE
command
 ? CATALOG         for help on all of the CATALOG commands.
```

```
To exit db2 interactive mode, type QUIT at the command prompt. Outside

interactive mode, all commands must be prefixed with 'db2'.

To list the current command option settings, type LIST COMMAND OPTIONS.

For more detailed help, refer to the Online Reference Manual.

db2 =>
```

```
db2 => create table db2ha.failover_test(month
varchar(10),year integer)
```

```
DB20000I  The SQL command completed successfully.
```

- Control the existence of the table:

```
db2 => list tables for schema db2ha show detail
```

```
Table/View                  Schema        Type  Creation time

------------------------------ --------------- ----- --------------------------

FAILOVER_TEST         DB2HA          T     2003-07-18-06.40.42.852982


  1 record(s) selected.
```

Since we created the table ‘failover_test’, we can cause the primary to fail so that the secondary can gain access to the shared file system:

```
db2 => quit

db2inst1@ha01:/root> exit

ha01:~ # shutdown –fr now
```

Step 2: Here we will insert some values into the table created above and cause the secondary node to fail in order to let the primary node get the database back.

- Login to the secondary node as db2 instance user and check the shared folder /db2ha for the database:

```
ha02:~ # su db2inst1
```

```
db2inst1@ha02:/root> ls  /db2ha
```

db2 **db2inst1 lost+found**

- Connect to the database db2ha:

```
db2inst1@ha02:/root> db2 connect to db2ha
```

Database Connection Information

Database server = DB2/LINUX 8.1.2

SQL authorization ID = DB2INST1
Local database alias = DB2HA

- Check the status of the database:

```
db2inst1@ha02:/root> db2 list active databases
```

Active Databases

Database name = DB2HA
Applications connected currently = 1
Database path = /db2ha/db2inst1/NODE0000/SQL00001/

- Enter the DB2 command line processor and insert some values into the fields (month, year) of the table ‘failover_test’.

```
db2inst1@ha02:/root> db2

db2 => insert into db2ha.failover_test (month, year)
values ('July',2003)
```

DB20000I The SQL command completed successfully.

- Now we can let the primary node take the database back by causing the secondary to fail.

```
db2 => quit

db2inst1@ha02:/root> exit

ha02:~ # shutdown -fr now
```

Step 3: Checking the database db2ha for the modified table on the primary node.

- Log in to the secondary node as the db2 instance user. After connecting to the database 'db2ha', enter the Db2 command line processor to check the contents of the table 'failover_test'

```
ha01:~ # su db2inst1

db2inst1@ha01:/root> db2 connect to db2ha
```

```
  Database Connection Information

 Database server        = DB2/LINUX 8.1.2
 SQL authorization ID   = DB2INST1
 Local database alias   = DB2HA
```

```
db2inst1@ha01:/root>db2
```

- Check the content of the table:

```
db2 => select * from db2ha.failover_test
```

```
MONTH      YEAR
---------- -----------
July            2003

  1 record(s) selected.
```

If you see this result, you have successfully completed the test.

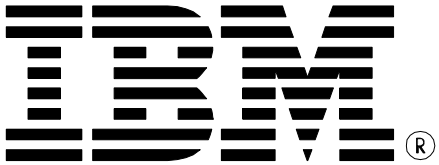
10. SUMMARY

As a result of industry growth, companies worldwide are becoming increasingly dependent on their IT systems.

No matter what their computer systems are, they all have a common need: a database environment to store all types of data. Data is the most important source for the company, thus it should be well-protected. All over the world, nearly 75% of mission-critical database applications experience nine hours of downtime per year. This means that most companies are under-investing in Linux High Availability technology even though they lose millions of dollars per hour due to downtime. As illustrated in this step-by-step guide, the Linux HA is easy to install and operate even for the end user who wants to secure the DB2 database environment.
By choosing IBM DB2, the best scaling database on Linux and by setting up Linux High Availability environment, companies can save not only time and money but have a secure business backend as well.

11. ACKNOWLEDGEMENTS

I would like to acknowledge the support and suggestions of ***Doug Rothert*** from the IBM Linux Integration Center and ***Alan Robertson*** from the IBM Linux Technology Center during the technical work for this paper.



IBM Canada
8200 Warden Avenue
Markham, ON
L6G 1C7
Canada

Printed in United States of America
8-03