By Francis
Published: 2007-08-16 17:09

Virtual Machine Replication & Failover with VMWare Server & Debian Etch (4.0)

Version 1.0
Author: Francis Theys <francis [at] fusiontek [dot] be>, Falko Timme <ft [at] falkotimme [dot] com>, Till Brehm <t [dot] brehm [at] projektfarm [dot] de>
Last edited 08/10/2007

This tutorial provides step-by-step instructions about how to create a highly available VMware Server environment on a Debian Etch system. With this tutorial, you will be able to create Virtual Machines that will be available on multiple systems with failover/failback capabilities.

The system is based using components of "**The High Availability Linux Project**" , namely "**DRBD**" and "**Heartbeat**".
The free open-source edition of DRBD will only allow a 2-node active/passive environment, so this is not for large businesses!. Also, the heartbeat/drbd setup configured in this tutorial, is by using 2 Ethernet NIC's. I recommend that at least the nic to be used for DRBD replication (eth1 in this tutorial) is 1Gbit or more.

Warning: ***This tutorial requires at least basic Linux & Networking knowledge and is not for so called "newbies" to Linux and/or Networking and/or Vmware !***

There are many ways of achieving this goal but this is the way I take. I do not issue any guarantee that this will work for you!

1. Preliminary Note

I assume you have already set up a basic Debian Etch system with VMWare Server installed on both servers as described in the following tutorials:

"***The Perfect Setup - Debian Etch (Debian 4.0)***" !!Only the first 2 pages!!

NOTE:When using a server with 4GB RAM or more, please install a "bigmem" kernel and respective headers!

```
apt-get install linux-image-2.6.18-4-686-bigmem linux-headers-2.6.18.-4-686-bigmem
```

Since we will be needing a separate filesystem to be replicated between both servers, I suggest using a manual partition scheme like the following:

```
/dev/sda1 -- 500 MB /boot (primary, ext3, Bootable flag: on)
/dev/sda5 -- 10000 MB / (logical, ext3)
/dev/sda6 -- 2000 MB swap (logical)
/dev/sda7 -- 60 GB unmounted (logical) (will contain the /var/vm directory)
```

You can vary the sizes of the partitions depending on your hard disk size, and the names of your partition might also vary, depending on your hardware (e.g. you might have /dev/hda1 instead of /dev/sda1 and so on) Also, make sure /dev/sda7 is identical in size on server1 and server2, and please do not mount them when the installer asks you:
No mount point is assigned for the ext3 file system in partition #7 of SCSI1 (0,0,0) (sda).
Do you want to return to the partitioning menu?
please answer No!

"***How To Install VMware Server On Debian 4.0***" !!Only the first page!!

UNLESS OTHERWISE SPECIFIED, ALL COMMANDS ARE DONE ON BOTH SERVERS

2. Installing the required packages on both servers

First we update our package database:

```
apt-get update
```

Now we install the packages required for this tutorial

```
apt-get install ssh drbd0.7* module-assistant heartbeat build-essential psmisc
```

3. Configuring DRBD and creating the replicated filesystem

When installing the drbd0.7 package, only the required module source package is copied to the /usr/src directory. To actually install and configure DRBD you will have to "make" it.

```
cd /usr/src
tar xzf drbd0.7.tar.gz
cd /usr/src/modules/drbd/drbd
make && make install
```

Note: If you get this error: ***"SORRY, kernel makefile not found. You need to tell me a correct KDIR!"*** then reboot first !

Now we need to configure DRBD to use our separate partition (/dev/sda7) as a DRBD device and then create a filesystem on it.
I suggest moving/renaming the installed drbd.conf and putting our own file in place

```
mv /etc/drbd.conf /etc/drbd.conf-sample
nano /etc/drbd.conf
```

You can use this `drbd.conf` file as a template:

```
resource vm1 {
  protocol C;
  incon-degr-cmd "echo '!DRBD! pri on incon-degr' | wall ; sleep 60 ; halt -f";
  startup {
    wfc-timeout  10;   # 10 seconds
    degr-wfc-timeout 30;    # 30 seconds
  }
  disk {
    on-io-error   detach;
  }
  net {
    max-buffers   20000;  # Play with this setting to achieve highest possible performance
    unplug-watermark   12000;         # Play with this setting to achieve highest possible performance
    max-epoch-size 20000;             # Should be the same as max-buffers
  }
```

```
 syncer {
  rate 10M;   # Use more if you have a Gigabit network. Speed is in Kylobytes. e.g.: 10M = 10Megabytes
  group 1;
  al-extents 257;
 }
 on server1 {  # Use the EXACT hostname of your server as give by the command "uname -n"
  device     /dev/drbd0;  # drbd device ID
  disk       /dev/sda7;   # physical disk device , check your partitioning scheme !!
  address    172.20.20.100:7789; # Fixed IP address of server1
  meta-disk  internal;  # I use internal metadata storage
 }
 on server2 {
  device     /dev/drbd0;
  disk       /dev/sda7;
  address    172.20.20.200:7789;
  meta-disk  internal;
 }
}
```

NOTE:THIS FILE MUST BE THE SAME ON BOTH SERVERS !

Now we can start the DRBD device and create the filesystem.

On both servers:

```
modprobe drbd
drbdadm up all
```

Now we define "server1" as the primary/master server:

On server1:

```
drbdsetup /dev/drbd0 primary --do-what-I-say
mkfs.ext3 /dev/drbd0
```

Wait a while to have the "ext3" filesystem created on /dev/drbd0 and then:

```
drbdadm connect all
```

And wait for the initial synchronisation to complete. On slower networks, this might take up to a few hours depending on the disksize! You can check the status of the synch with this command:

```
cat /proc/drbd
```

Which should give you an output during synch similar to this:

```
version: 0.7.10 (api:77/proto:74)SVN Revision: 1743 build by phil@mescal, 2005-01-31 12:22:07
0: cs:SyncSource st:Primary/Secondary ld:Consistent
     ns:13441632 nr:0 dw:0 dr:13467108 al:0 bm:2369 lo:0 pe:23 ua:226 ap:0
     [==========>.........] sync'ed: 53.1% (11606/24733)M
     finish: 1:14:16 speed: 2,644 (2,204) K/sec
1: cs:Unconfigured
```

NOTE: Your diskwrite performance will be limited to the synch speed you see here !! Check your buffer size to increase this up to optimal values! (you can make config changes and then perform: '/etc/init.d/drbd reload')

Check the status periodically until it is completed, which should give output similar to this:

```
SVN Revision: 1743 build by phil@mescal, 2005-01-31 12:22:07
0: cs:Connected st:Primary/Secondary ld:Consistent
```

```
   ns:37139 nr:0 dw:0 dr:49035 al:0 bm:6 lo:0 pe:0 ua:0 ap:0
1: cs:Unconfigured
```

When the synch is complete, it is time to mount our drbd filesystem on to the previously created "/var/vm" directory as specified for the Virtual Machines during the installation of VMWare Server.

```
mount -t ext3 /dev/drbd0 /var/vm
```

This part of the tutorial concludes the volume replication of your servers, that will enable the Virtual Machines to be replicated on to both servers. This allows for data security and makes sure that virtual machines created on 1 server will always be available on both servers. You should now create your Virtual Machines that you want to have in your failover. Please check page 2 of: "***How To Install VMware Server On Debian 4.0***" for more information on how to do this. You will need the VM's name and configfile name to proceed !

The next part involves configuring the HeartBeat package and making sure that in case of failover, the virtual machines are properly initialized and started on the secondary server.

4. Configuring the Heartbeat package

Installing the Heartbeat package will enable automatic failover and resource management in case of resource and/or hardware failure.

In this tutorial, I will use Heartbeat over the network, but it is also possible to use Heartbeat with a serial cable and other methods. Heartbeat will check for updates from both servers, and when not receiving any more updates will initiate a failover and takeover of the specified resources (in our case, DRBD and VMWare).

UNLESS OTHERWISE SPECIFIED, ALL COMMANDS ARE DONE ON BOTH SERVERS

OK, on to the configuration. For your convenience, you can use the following configuration files as they are tailored to this tutorial. Heartbeat needs 3 files to be configured, namely: "ha.cf", "haresources" and "authkeys". Edit them accordingly with your favorite editor:

```
nano /etc/ha.d/ha.cf
```

```
logfile /var/log/ha-log
logfacility local0
keepalive 1
deadtime 10
warntime 10
udpport 694
bcast eth1
auto_failback off
node server1  ## make sure both names are accessible - check /etc/hosts
node server2
ping 192.168.0.254  ## Enter an IP address that is pingable from the ETH0 network !!
respawn hacluster /usr/lib/heartbeat/ipfail
```

NOTE: This ha.cf file will enable failover in case of Network Connection failure on ETH0 and/or general server failure. Problems with ETH1 will <u>NOT</u> cause a failover to occur !

Now we edit the "authkeys" file:

```
nano /etc/ha.d/authkeys
```

```
auth 3
3 md5 failover  ## this is just a string, enter what you want ! auth 3 md5 uses md5 encryption
```

But we also have to chmod this file to only be readable by "root":

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

And for last, we edit the "haresources" file that contains the failover resources.


```
nano /etc/ha.d/haresources
```

```
server1 192.168.0.10 drbddisk::vm1 Filesystem::/dev/drbd0::/var/vm::ext3 vmstart
# Explanation:
# Primary Server name --> virtual IP address to be used --> DRBD resource as configurd in /etc/drbd.conf
# --> where to mount the DRBD resource and the filesystem type --> resource to start/stop in case of failover
# the "vmstart" resource is a custom script needed for VMWare Server since you can't have heartbeat take control of
# the vmware services or you will run into the dreaded "not configured for this machine" errors !
```

Before going further, we will need to create the custom "vmstart" script to make sure that Virtual Machines are automatically started on the server in case of failover. With other resources like Apache, Mysql, NFS ... we could have simply removed the startup scripts for them from the rc.d directories, but with VMWare this is not possible as they are required for good functioning of the services. Therefore, we need a custom script to start the Virtual Machines in case of failover since the VMWare services will already be running prior to failover !

The "vmstart" will need to reside in the "/etc/ha.d/resource.d/" folder as that is where heartbeat will check for it.

```
nano /etc/ha.d/resource.d/vmstart
```

```
#!/bin/bash
case "$1" in
start)
/usr/bin/vmware-cmd -s register "/var/vm/'virtual machine name'/'vm config file'.vmx" 2>/dev/null
/usr/bin/vmware-cmd "/var/vm/'virtual machine name'/'vm config file'.vmx" start 2>/dev/null
;;
stop)
/usr/bin/vmware-cmd "/var/vm/'virtual machine name'/'vm config file'.vmx" stop trysoft 2> /dev/null
;;
status)
if `/usr/bin/vmware-cmd "/var/vm/'virtual machine name'/'vm config file'.vmx" getstate 2>/dev/null | grep -q "getstate() = on"`
then
```

```
echo "running"
else
echo "stopped"
fi
;;
*)
echo "Usage: `basename $0` 'virtual machine file' {start|stop|status}"
echo "Where "virtual machine file" is like /var/vm/'virtual machine name'/'vm config file'.vmx"
;;
esac
exit 0
```

Fill in the 'virtual machine' name and 'vm config file' directives with the necessary info (check the */var/vm* dir). If you have multiple Virtual Machines, just copy the lines.

This custom script will first Register the Virtual Machine (necessary when using a newly installed server!) and then start them. It has to be done this way, since VMWare Server only has the option to automatically start VM's on system startup.
Now that you have configured everything, it is time to test if all is OK. Start the Heartbeat service, first on Server1, then on Server2.

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

If all is OK, you should see the new virtual IP showing up on your master server.

```
ifconfig |more
```

Output:

```
eth0 Link encap:Ethernet  HWaddr 00:11:09:00:BB:5D
          inet addr:192.168.0.1  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::211:9ff:fe00:bb5d/64 Scope:Link
```



```
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1025774 errors:0 dropped:0 overruns:0 frame:0
          TX packets:227653 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:281072399 (268.0 MiB)  TX bytes:56403081 (53.7 MiB)
eth0:0    Link encap:Ethernet  HWaddr 00:11:09:00:BB:5D
          inet addr:192.168.0.10  Bcast:192.168.0.255  Mask:255.255.255.0
          ...
```

Ok, if you are feeling lucky and have followed this tutorial with some common sense, then it is now time to yank out that power cord, hold on to your cup of coffee and watch the Virtual Machine go online on your second server.

5. Links

- Debian: http://www.debian.org
- The High Availability Linux Project: http://www.linux-ha.org
- DRBD: http://www.drbd.org
- VMware Server: http://www.vmware.com/products/server
- VMware Appliance Directory http://www.vmware.com/vmtn/appliances/