Xen Cluster Management With Ganeti On Debian Lenny

Version 1.0
Author: Falko Timme

On this page

Ganeti is a cluster virtualization management system based on Xen. In this tutorial I will explain how to create one virtual Xen machine (called an `instance`) on a cluster of two physical nodes, and how to manage and failover this instance between the two physical nodes.

This document comes without warranty of any kind! I do not issue any guarantee that this will work for you!

[Update 01/21/2010] I got a message from the Ganeti development team:

"*[...] In recent months we noticed the unfortunate fact that people try to follow your instructions to the letter and end up installing old or very old versions of Ganeti. Could you please update both tutorials with notes saying that they aren't updated for more recent Ganeti versions and ask people to look at the up-to-date documentation on http://docs.ganeti.org/ganeti/?*"

This tutorial is based on an old version of Ganeti. Please refer to the up-to-date documentation on http://docs.ganeti.org/ganeti/.

1 Preliminary Note

In this tutorial I will use the physical nodes `node1.example.com` and `node2.example.com`:

- `node1.example.com`: IP address `192.168.0.100`; will be the master of the cluster.
- `node2.example.com`: IP address `192.168.0.101`; will be the primary node of the virtual machine (aka `instance`).

Both have a 500GB hard drive of which I use 20GB for the `/` partition, 1GB for swap, and leave the rest unpartitioned so that it can be used by Ganeti (the minimum is 20GB!). Of course, you can change the partitioning to your liking, but remember about the minimum unused space.

The cluster I'm going to create will be named `cluster1.example.com`, and it will have the IP address `192.168.0.102`. The cluster IP `192.168.0.102` will always be bound to the cluster master, so even if you don't know which node is the master, you can use the cluster IP (or the hostname `cluster1.example.com`) to connect to the master using SSH.

The Xen virtual machine (called an `instance` in Ganeti speak) will be named `inst1.example.com` with the IP address `192.168.0.105`. `inst1.example.com` will be mirrored between the two physical nodes using DRBD - you can see this as a kind of network RAID1.

As you see, `node1.example.com` will be the cluster master, i.e. the machine from which you can control and manage the cluster, and `node2.example.com` will be the primary node of `inst1.example.com`, i.e. `inst1.example.com` will run on `node2.example.com` (with all changes on `inst1.example.com` mirrored back to `node1.example.com` with DRBD) until you fail it over to `node1.example.com` (if you want to take down `node2.example.com` for maintenance, for example). This is an active-passive configuration.

I think it's good practice to split up the roles between the two nodes, so that you don't lose the cluster master and the primary node at once should one node go down.

It is important that all hostnames mentioned here should be resolvable to all hosts, which means that they must either exist in DNS, or you must put all hostnames in all */etc/hosts* files on all hosts (which is what I will do here).

All cluster nodes must use the same network interface (e.g. *eth0*). If one node uses *eth0* and the other one *eth1*, then Ganeti won't work correctly anymore.

Ok, let's start...

2 Preparing The Physical Nodes

node1:

I want *node1* to have the static IP address *192.168.0.100*, therefore my */etc/network/interfaces* file looks as follows (please note that I replace *allow-hotplug eth0* with *auto eth0*; otherwise restarting the network doesn't work, and we'd have to reboot the whole system):

```
vi /etc/network/interfaces
```

```
# The loopback network interface
auto lo
iface lo inet loopback
# The primary network interface
#allow-hotplug eth0
#iface eth0 inet dhcp
auto eth0
iface eth0 inet static
        address 192.168.0.100
        netmask 255.255.255.0
        network 192.168.0.0
        broadcast 192.168.0.255
        gateway 192.168.0.1
```

If you've modifed the file, restart your network:

```
/etc/init.d/networking restart
```

Then edit */etc/hosts*. Make it look like this:

```
vi /etc/hosts
```

```
127.0.0.1       localhost.localdomain   localhost
192.168.0.100   node1.example.com       node1
192.168.0.101   node2.example.com       node2
192.168.0.102   cluster1.example.com    cluster1
192.168.0.105   inst1.example.com       inst1
# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
```

Next we must make sure that the commands

```
hostname
```

and

```
hostname -f
```

print out the full hostname (*node1.example.com*). If you get something different (e.g. just *node1*), do this:

```
echo node1.example.com > /etc/hostname
/etc/init.d/hostname.sh start
```

Afterwards, the *hostname* commands should show the full hostname.

Then update the system:

```
aptitude update
```

```
aptitude safe-upgrade
```

node2:

Now we do the same again on *node2.example.com* (please keep in mind that *node2* has a different IP!):

```
vi /etc/network/interfaces
```

```
# The loopback network interface
auto lo
iface lo inet loopback
# The primary network interface
#allow-hotplug eth0
#iface eth0 inet dhcp
auto eth0
iface eth0 inet static
        address 192.168.0.101
        netmask 255.255.255.0
        network 192.168.0.0
        broadcast 192.168.0.255
        gateway 192.168.0.1
```

```
/etc/init.d/networking restart
```

```
vi /etc/hosts
```

```
127.0.0.1       localhost.localdomain   localhost
192.168.0.100   node1.example.com       node1
192.168.0.101   node2.example.com       node2
192.168.0.102   cluster1.example.com    cluster1
192.168.0.105   inst1.example.com       inst1
# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
```

```
echo node2.example.com > /etc/hostname
/etc/init.d/hostname.sh start
```

```
aptitude update
```

```
aptitude safe-upgrade
```

3 Setting Up LVM On The Free HDD Space

node1/node2:

Let's find out about our hard drive:

```
fdisk -l
```

```
node1:~# fdisk -l

Disk /dev/sda: 500.1 GB, 500107862016 bytes
255 heads, 63 sectors/track, 60801 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Disk identifier: 0x00023cd1

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          62      497983+  83  Linux
/dev/sda2              63        6141    48829567+  8e  Linux LVM
node1:~#
```

We will now create the partition */dev/sda3* (on both physical nodes) using the rest of the hard drive and prepare it for LVM:

```
fdisk /dev/sda
```

```
node1:~# fdisk /dev/sda

The number of cylinders for this disk is set to 60801.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): <-- n
Command action
   e   extended
   p   primary partition (1-4)
<-- p
Partition number (1-4): <-- 3
First cylinder (6142-60801, default 6142): <-- ENTER
Using default value 6142
Last cylinder or +size or +sizeM or +sizeK (6142-60801, default 60801): <-- ENTER
Using default value 60801

Command (m for help): <-- t
Partition number (1-4): <-- 3
Hex code (type L to list codes): <-- L

 0  Empty           1e  Hidden W95 FAT1 80  Old Minix       be  Solaris boot
 1  FAT12           24  NEC DOS         81  Minix / old Lin bf  Solaris
 2  XENIX root      39  Plan 9          82  Linux swap / So c1  DRDOS/sec (FAT-
 3  XENIX usr       3c  PartitionMagic  83  Linux           c4  DRDOS/sec (FAT-
 4  FAT16 <32M      40  Venix 80286     84  OS/2 hidden C:  c6  DRDOS/sec (FAT-
 5  Extended        41  PPC PReP Boot   85  Linux extended  c7  Syrinx
 6  FAT16           42  SFS             86  NTFS volume set da  Non-FS data
 7  HPFS/NTFS       4d  QNX4.x          87  NTFS volume set db  CP/M / CTOS / .
 8  AIX             4e  QNX4.x 2nd part 88  Linux plaintext de  Dell Utility
 9  AIX bootable    4f  QNX4.x 3rd part 8e  Linux LVM       df  BootIt
 a  OS/2 Boot Manag 50  OnTrack DM      93  Amoeba          e1  DOS access
 b  W95 FAT32       51  OnTrack DM6 Aux 94  Amoeba BBT      e3  DOS R/O
```

```
 c  W95 FAT32 (LBA) 52  CP/M            9f  BSD/OS          e4  SpeedStor
 e  W95 FAT16 (LBA) 53  OnTrack DM6 Aux a0  IBM Thinkpad hi eb  BeOS fs
 f  W95 Ext'd (LBA) 54  OnTrackDM6      a5  FreeBSD         ee  EFI GPT
10  OPUS            55  EZ-Drive        a6  OpenBSD         ef  EFI (FAT-12/16/
11  Hidden FAT12    56  Golden Bow      a7  NeXTSTEP        f0  Linux/PA-RISC b
12  Compaq diagnost 5c  Priam Edisk     a8  Darwin UFS      f1  SpeedStor
14  Hidden FAT16 <3 61  SpeedStor       a9  NetBSD          f4  SpeedStor
16  Hidden FAT16    63  GNU HURD or Sys ab  Darwin boot     f2  DOS secondary
17  Hidden HPFS/NTF 64  Novell Netware  b7  BSDI fs         fd  Linux raid auto
18  AST SmartSleep  65  Novell Netware  b8  BSDI swap       fe  LANstep
1b  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid ff  BBT
1c  Hidden W95 FAT3 75  PC/IX
Hex code (type L to list codes): <-- 8e
Changed system type of partition 3 to 8e (Linux LVM)

Command (m for help): <-- w
The partition table has been altered!

Calling ioctl() to re-read partition table.

WARNING: Re-reading the partition table failed with error 16: Device or resource busy.
The kernel still uses the old table.
The new table will be used at the next reboot.
Syncing disks.
node1:~#
```

Now let's take a look at our hard drive again:

```
fdisk -l
```

```
node1:~# fdisk -l

Disk /dev/sda: 500.1 GB, 500107862016 bytes
255 heads, 63 sectors/track, 60801 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Disk identifier: 0x00023cd1

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          62      497983+  83  Linux
/dev/sda2              63        6141    48829567+  8e  Linux LVM
/dev/sda3            6142       60801   439056450   8e  Linux LVM
node1:~#
```

Looks good. Now we must reboot both physical nodes so that the kernel can read in the new partition table:

```
reboot
```

After the reboot, we install LVM (probably it's already installed, but it's better to go sure):

```
aptitude install lvm2
```

After the reboot, we prepare */dev/sda3* for LVM on both nodes and add it to the volume group *xenvg*:

```
pvcreate /dev/sda3
vgcreate xenvg /dev/sda3
```

(Ganeti wants to use a volume group of its own, that's why we create *xenvg*; theoretically we could use an existing volume group with enough unallocated space, but the *gnt-cluster verify* command will complain about this.)

Next >>

Sub pages

Xen Cluster Management With Ganeti On Debian Lenny
Xen Cluster Management With Ganeti On Debian Lenny - Page 2
Xen Cluster Management With Ganeti On Debian Lenny - Page 3
Xen Cluster Management With Ganeti On Debian Lenny - Page 4

Suggested articles

0 Comment(s)

Add comment

Name *

Email *

p

I'm not a robot reCAPTCHA Privacy - Terms

Submit comment

Comments

From: falko Reply

Tutorial Info

Author: falko
Tags: virtualization, debian, high-availability, xen

Share This Page