

Praktischer Teil – Kurzanleitung

Diese Kurzanleitung ist kein “Walk-Through”-Handbuch, sondern soll lediglich einen Rahmen für die Clusterinbetriebnahme mit VMware zeigen.

Kommandozeilenaufrufe und Dateiinhalte sind in der Schriftart `Courier` dargestellt, Hostnamen, Verzeichnis- und Dateinamen in *Arial kursiv*.

Beispiele:

Hostname: *host1*

Dateiname: */etc/cluster/cluster.conf*

Kommandozeilen-Operation: `vmware-console`

Vmware Server

Auf allen Arbeitsplatzrechnern ist Vmware Server (ideal: Version ≥ 1.02) (<http://www.vmware.com/download/server/>) auf einem Linuxbetriebssystem installiert. Vmware ermöglicht durch Virtualisierung, in einer virtuellen PC-Umgebung mehrere voneinander unabhängige Betriebssysteme/virtuelle Server zu betreiben. Dies ist kein Standardszenario für den Einsatz von GFS, sondern dient nur als einfaches Setup für Workshopumgebungen.

Es müssen zwei bzw. drei virtuelle Maschinen über die `vmware-console` angelegt werden, die auf eine (oder mehrere) gesharete Disks zugreifen können. Auf diesen Disks wird dann jeweils ein GFS Filesystem aufgesetzt. Das gemeinsam genutzte Dateisystem wird dann verwendet, um einen so genannten Diskless Shared Root Cluster zu formen. Diese spezielle Cluster-Art verzichtet auf lokale Festplatten und bootet direkt von einem Shared Storage in ein Single System Image. Die Cluster-Knoten verbinden sich beim Booten automatisch und teilen sich eine gemeinsame Root-Partition. Egal, wie viele Knoten zu einem Cluster verbunden werden, das System lässt sich wie ein einzelner Linux Server konfigurieren.

Installation von CentOS als virtuelle Maschine

Legen Sie in der `vmware-console` eine neue virtuelle Maschine an. Mounten sie als

Installationsquelle hierzu das entsprechende CentOS Linux ISO-Image (z.B. die CentOS 4.4 Server ISO) und führen Sie die Installation **einer** virtuellen Maschine aus. Ist dies geschehen, kann durch die Eingabe von `yum update` auf der Kommandozeile die Installation auf den aktuellsten Patchstand gebracht werden.

Bitte richten Sie die virtuelle Maschine im *Inventory-Fenster (Rechte Maustaste->New>Virtual Machine)* der `vmware-console` mit folgenden Eckparametern ein:

Virtual Machine Configuration: *Typical*
Guest Operating System: *Red Hat Enterprise Linux 4*
Name: *host1*
Location: */var/lib/vmware/Virtual Machines/host1/*
Network Connection: *Use Network Address Translation (NAT)*
Disk Size: *3 GB*
Allocate Disk Space now: *NEIN* (bzw. Häkchen entfernen!)

Hinweis:

CentOS 4.4 ist ein freier RHEL 4 Clone, kann allerdings im Gegensatz zu einer entsprechend subscriperten RHEL Version nicht sofort via `up2date` aus den RedHat Network Repositories upgedated werden.

Die Maschine hat per DHCP eine IP Adresse zugewiesen bekommen, bitte ändern Sie die Einstellung in eine feste IP aus dem selben Class-C Netz. Die Adressen sind nur auf dem Vmware-Host bekannt, Inkonsistenzen mit benachbarten Systemen können also nicht auftreten.

Merken Sie sich die zuvor per DHCP vergebene Gatewayadresse (meistens *.*.*.2) und setzen Sie diese als manuelles Gateway, ansonsten ist keine Kommunikation nach aussen mehr möglich.

Sie können die benötigten Änderungen mit dem Befehl `system-config-network` vornehmen. Setzen Sie zusätzlich bitte den korrekten Hostnamen in `/etc/sysconfig/network`.

Derselbe Name wird auch im nächsten Schritt in die Datei `/etc/hosts` eingetragen.

Editieren Sie die Datei `/etc/hosts`, um allen am Cluster beteiligten Rechnern einen gültigen

Eintrag zu geben. Bitte tragen Sie auch die für den Testaufbau geplanten weiteren Node-IPs (*host2*, *host3*) ein, sowie den Host, auf dem die Vmware läuft.

```
<ip.erste.virt.maschine>      host1
<ip.zweite.virt.maschine>     host2
<ip.dritte.virt.maschine>     host3
<ip.des.vmware.host>         vmware-host
```

Beispielsweise können folgende IP Adressen verwendet werden:

```
172.16.40.50      host1
172.16.40.51      host2
172.16.40.52      host3
172.16.40.100     vmware-host
```

Rebooten Sie die virtuelle Maschine im Anschluss.

Installation der GFS und com.oonics Pakete

Erstellen Sie die Datei */etc/yum.repos.d/CentOS-csgfs.repo* mit folgendem Inhalt:

```
[csgfs]
name=CentOS-4 - CSGFS
baseurl=http://mirror.centos.org/centos/$releasever/csgfs/$basearch/
gpgcheck=1
enabled=1
```

Ebenfalls muss die Datei */etc/yum.repos.d/comoonics.repo* erstellt werden, die die com.oonics Quellen enthält:

```
[comoonics-noarch]
name=Packages for the comoonics shared root cluster
baseurl=http://download.atix.de/yum/comoonics/productive/noarch/
enabled=1
gpgcheck=1
gpgkey=http://download.atix.de/yum/comoonics/comoonics-RPM-GPG.key
```

```
[comoonics]
name=Packages for the comoonics shared root cluster
baseurl=http://download.atix.de/yum/comoonics/productive/$basearch/
enabled=1
gpgcheck=1
gpgkey=http://download.atix.de/yum/comoonics/comoonics-RPM-GPG.key
```

Auf Kommandozeile müssen jetzt die GFS- und alle weiteren Clusterpakete installiert werden. Diese werden u.a. aus dem eben hinzugefügten Repositories nebst Abhängigkeiten heruntergeladen.

Zunächst wird GFS und die Cluster Suite installiert:

```
yum install GFS-6*
yum install GFS-kernel-2*

yum install magma-1.0*
yum install magma-devel-1.0*
yum install magma-plugins-1.0*

yum install dlm-1.0.*
yum install fence-1.32*

yum install system-config-cluster
yum install lvm2-cluster

yum install perl-Net-Telnet*
```

Kontrollieren Sie bitte, ob folgende Zeilen in der Datei */etc/lvm/lvm.conf* enthalten sind. Diese Zeilen sind wichtig, damit LVM2 clusterkompatibel arbeiten kann.

```
locking_type = 1
locking_library = "liblvm2clusterlock.so"
```

Wichtig: Für die nachfolgende Installation muss *tmpwatch* deinstalliert werden:

```
rpm -e tmpwatch
```

Nun müssen die *com.oonics* Pakete mit *yum* installiert werden:

```
yum install PyXML
yum install libxslt-python
yum install perl-libxml-errno

yum install comoonics-bootimage-1.0*
yum install comoonics-bootimage-fenceacksv
yum install comoonics-cs*
yum install comoonics-ec*
```

Passwortlose Authentifikation mittels ssh

Optional können Vmware-Agents für das Fencing benutzt werden. Diese benötigen `ssh` zur Ausführung von Remotebefehlen. Hierzu müssen die Clients sowie der *vmware-host* entsprechend konfiguriert werden.

Auf der virtuellen Maschine folgenden Befehl ausführen: `ssh-keygen -t dsa`
Wird nach einem Passwort gefragt, die Abfrage ohne Passworteingabe bestätigen.

Es werden in `/root/.ssh/` zwei Dateien (*id_dsa*, *id_dsa.pub*) angelegt.
Der Inhalt der Datei *id_dsa.pub* muß auf *vmware-host* an die Datei `/root/.ssh/authorized.keys` angefügt werden.

Fencing der Cluster-Nodes

Um die einzelnen virtuellen Maschinen der Clusternodes später korrekt fencen zu können, benötigen Sie spezielle fencing Agents fuer Vmware, andernfalls ist nur manuelles "Pseudo-"Fencing möglich.

Die notwendigen Agents und Installationsbeschreibungen finden Sie im Downloadbereich auf <http://open-sharedroot.org>. Laden Sie sich diese bitte z.B. mittels des Kommandos `wget` herunter.

Bitte beachten Sie die Release Notes, da neue VmWare Versionen mitunter andere Agents benötigen.

Auf *vmware-host*: *fence_vmware_master*

Legen Sie das Script unter `/opt/scripts/` ab und geben Sie dem User *root* Executerechte.

Auf *host1*: *fence_vmware_client*

Legen Sie das Script unter `/opt/scripts/` ab und geben Sie dem User *root* Executerechte.

Folgen Sie bitte den Installationsanleitungen der beiden Scripts. Passen Sie ggf. Die Pfade an (notwendig im Client-Script).

Configuration des Clusters

Wichtigste Konfigurationsdatei für den Cluster ist die Datei `/etc/cluster/cluster.conf` mit folgendem Inhalt:

```
<?xml version="1.0"?>
<cluster config_version="1" name="decuscluster">
  <fence_daemon clean_start="1" post_fail_delay="0" post_join_delay="3"/>
  <clusternodes>
    <clusternode name="host1" votes="1" nodeid="1">
      <fence>
        <method name="1">
          <device name="fence_manual" nodename="host1"/>
        </method>
      </fence>
    </clusternode>
    <clusternode name="host2" votes="1" nodeid="2">
      <fence>
        <method name="1">
          <device name="fence_manual" nodename="host2"/>
        </method>
      </fence>
    </clusternode>
  </clusternodes>
  <cman expected_votes="1" two_node="1"/>
  <fencedevices>
    <fencedevice agent="fence_manual" name="fence_manual"/>
  </fencedevices>
</cluster>
```

Sie können die Datei entweder per Texteditor (zB `vi`) oder mittels grafischem Konfigurationstool erstellen (`system-config-cluster`). Der einfachste Weg ist es, die initiale Konfiguration mittels `system-config-cluster` zu erzeugen und Detailänderungen per Texteditor durchzuführen.

Um `system-config-cluster` ausführen zu können, muss allerdings X auf dem Cluster-Knoten installiert sein. Aus Platzgründen wurde darauf bei den vorbereiteten Images verzichtet und die Datei ist aus der Vorlage zu erstellen.

Zu beachten ist auch, dass es sich hierbei um eine rudimentäre `Cluster.conf` handelt, die später um weitere Funktionalität erweitert werden muss.

Shared GFS Disk für /boot und gemeinsames root anlegen

Fahren Sie die virtuelle Maschine herunter und gehen Sie in die Settings der virtuellen Maschine.

Legen Sie hier zwei neue Disks an (*Add->Harddisk*). Geben Sie einer Disk im *Advanced Menü* die *SCSI-ID 1:0*, der anderen die *SCSI-ID 1.1*. Die Disk für */boot* sollte mindestens 500MB groß sein, um ausreichend Platz für die spezielle *com.oonics Initrd* zur Verfügung zu haben. Die Disk für das *Shared Root* sollte mindestens 2GB gross sein. Es kann sich in beiden Fällen um eine dynamische Disk (der *phys. Speicherplatz* auf *vmware-host* wird nicht sofort vorbelegt) handeln, diese lassen sich schneller anlegen.

Als Speicherort sollten Sie einen eigenen Ordner auswählen (z.B. *shared-boot* und *shared-root* im Unterverzeichnis von */var/lib/vmware/Virtual Machines/*)

Gehen Sie nun in das Verzeichnis */var/lib/vmware/Virtual Machines/host1/* und öffnen Sie die Datei *host1.vmx*.

Fügen Sie die Einträge:

```
#SCSI Bus komplett sharen  
scsil.sharedBus = "virtual"
```

```
#DISK Locking DER GESAMTEN VIRTUELLEN MASCHINE abschalten  
disk.locking = "false"
```

hinzu.

Nach einem Reboot der virtuellen Maschine lassen Sie bitte im Bootscreen die Hardwareerkennung durchlaufen und bestätigen Sie den neu erkannten LSI-Logic Controller.

Hinweis:

Sollte die Maschine nicht booten, sondern im *fence*-Dämonen hängen bleiben, den Bootvorgang bitte im "Interactive"-Mode durchführen und den Start des *fence*-Dämonen nicht zulassen. Üblicherweise muß die "Interactive"-Methode bei einem Cluster mit mehr als zwei Knoten angewandt werden.

Nach erfolgreichem Reboot muss jetzt eine Volume Group samt Logical Volume auf der geshareten Disk angelegt werden. Des Weiteren muss ein neues Boot Volume für den Diskless Shared Root Cluster erstellt werden.

Die neuen Disks sollten nach dem Boot korrekt als Device *sdb* bzw. *sdc* erkannt werden. Dies kann mittels folgenden Befehlen geprüft werden:

```
cat /proc/scsi/scsi
cat /proc/partitions
fdisk -l
```

Einrichtung der weiteren Clusternodes

Bevor das GFS Filesystem auf *lv_shared-root* angelegt werden kann, muß erst ein gültiges (laufendes) Cluster vorhanden sein. Hierzu sind, je nach Konfiguration, eine bzw. zwei virtuelle Maschinen zu erstellen, die als einzige Platten die *shared-boot* und *shared-root* Disk erhalten.

Wichtig: Es ist zu beachten, dass wieder die **gleichen** SCSI-IDs verwendet und ebenfalls die speziellen Einstellungen in den *host*.vmx* vorgenommen werden!

Nur die erste virtuelle Maschine verfügt über lokale HDDs und dient als Grundlage, um ein gesharetes Root anzulegen.

Erstellen Sie auf dem *vmware-host* nun eine bzw. zwei zusätzliche Virtuelle Maschinen in entsprechenden Unterverzeichnissen (*/host2/*, */host3/*) des Verzeichnisses */var/lib/vmware/Virtual Machines/*.

Öffnen Sie die neue virtuelle Maschine im Inventory Tab der *vmware-console* und fügen Sie die geshareten Disks hinzu. (*Add->Disk->Use Existing Disk-><pfad zur geshareten Disk>*).

Geben Sie auch diesen Disks im *Advanced Menü* die *SCSI-ID 1:0* bzw. *SCSI-ID 1:1* und prüfen Sie ob danach die korrekten *shared-mode* Einträge in der **.vmx* Datei zu finden sind.

Booten Sie die neue virtuelle Maschine und geben Sie ihr (wie bereits fuer *host1* beschrieben) die korrekte feste IP samt Hostnamen. Übernehmen Sie bitte die Einträge der */etc/hosts* von *host1*.

Falls es beim Neustart des Netzwerkes Probleme wegen geänderter MAC-Adressen geben sollte, ändern Sie bitte in der Datei `/etc/sysconfig/network-scripts/ifcfg-eth0` (falls eth0 die korrekte Netzwerkkarte in Ihrem Setup ist) den Parameter `HWADDR` in die korrekte MAC-Adresse (abfragbar mittels `ifconfig eth0`). Nach einem Neustart des Netzwerkes mittels `/etc/init.d/network restart`, sollte das Netzwerk korrekt gestartet werden.

Falls Sie zwei neue Hosts angelegt haben, bitte alle Schritte für jeden einzelnen neuen virtuellen Server durchführen.

Booten Sie im Anschluss alle virtuellen Server, sie sollten jetzt ohne manuellen Eingriff hochfahren.

Prüfen Sie, ob alle zwei/drei Nodes Mitglied des Clusters sind. Ist dies nicht der Fall, liegt ein Fehler vor, es sollte im nächsten Schritt kein Filesystem angelegt werden.

Die notwendige Abfrage erfolgt mittels: `cman_tool status`

Beispielhafte Ausgabe:

```
Protocol version: 5.0.1
Config version: 15
Cluster name: demo_cluster
Cluster ID: 61876
Cluster Member: Yes
Membership state: Cluster-Member
Nodes: 3
Expected_votes: 3
Total_votes: 3
Quorum: 2
Active subsystems: 4
Node name: centos5-1.vmware.tn
Node addresses: 192.168.32.51
```

Disk und Partition für GFS Shared-Root Filesystem anlegen

Jetzt muss das LVM2-kompatible phys. Volume inkl. Partition für das Shared-Root darauf angelegt werden. Es wird dabei das GFS-Filesystem verwendet.

Ein phys. Volume erstellen:

```
pvcreate /dev/sdc
```

Eine Volume Group (*vg_shared-root*) erstellen:

```
vgcreate vg_shared-root /dev/sdc
```

Ein Logical Volume (*lv_shared-root* innerhalb der Volume Group erstellen:

```
lvcreate -n lv_shared-root -l xxx demo_vg
```

Mit dem Parameter `-l` wird dabei die Anzahl der Physical Extends übergeben. Hier kann man den Maximalwert wählen, den man sich über den Befehl ausgeben lassen kann:

```
vgdisplay | grep "Total PE" | awk '{print $3}'
```

Kontrolle, ob alles geklappt hat:

```
lvscan
```

LVM sollte von nun an nach dem Starten automatisch aktiviert sein.

Um LVM nachträglich zu aktivieren, `vgchange -ay` eingeben, alle aktiven Volume Groups werden gelistet.

GFS Filesystem anlegen

Jetzt kann das GFS-Filesystem auf dem geshareten Logical Volume angelegt werden.

Geben Sie hierzu folgende Kommandozeile ein:

```
gfs_mkfs -t decuscluster:lv_shared-root -j 3 -p lock_dlm  
/dev/vg_shared-root/lv_shared-root
```

Es wird standardmäßig eine Metajournal-Grösse von 128MB pro Clusterknoten (3x 128MB = 384 MB) erzeugt und vom zur Verfügung stehenden Speicherplatz des Logical Volumes abgezogen. Soll das Journal kleiner sein, muss mittels Kommandozeilenparameter `-J` eine andere Journalgrösse (mind. 32MB) angesetzt werden.

Geben Sie jetzt einen Mountpunkt vor (zb: `/cluster/mnt/vg_shared-root-lv_shared-root`) und mounten Sie das Logical Volume von einem Node aus. Der Mountvorgang erfolgt wie bei normalen Filesystemen:

```
mount -t gfs /dev/vg_shared-root/lv_shared-root
/cluster/mnt/vg_shared-root-lv_shared-root
```

Dieser Mountpoint wird nun schrittweise darauf vorbereitet, das komplette Shared Root Dateisystem abzubilden.

Shared Root erzeugen

Zunächst muss das lokal installierte Linux auf das GFS Volume kopiert werden:

```
cp -ax / /cluster/vg_shared-root-lv_shared-root/
```

Die Parameter `-ax` sorgen dafür, dass alles rekursiv kopiert wird, Links beibehalten werden und der Mountpoint nicht verlassen wird.

Als nächstes wird das neue, gesharete Root soweit vorbereitet, dass mit `chroot` in die neue Umgebung gewechselt werden kann. Hierzu sind folgende Mounts durchzuführen:

```
mount --bind /dev /cluster/mount/vg_shared-root-lv_shared-root/dev
```

```
mount -t proc none /cluster/mount/vg_shared-root-lv_shared-root/proc
```

```
mount -t sysfs none /cluster/mount/vg_shared-root-lv_shared-root/sys
```

Da der Shared Root Cluster tatsächlich für alle Knoten das selbe root Volume verwendet, die einzelnen Cluster-Knoten aber auch über spezielle, hostabhängige Einstellungen verfügen (z.B. die Netzwerkkarten-Konfiguration), müssen diese mit einer speziellen Verzeichnisstruktur verwaltet werden.

Symbolische Links stellen dabei sicher, dass man die hostabhängigen Datei an den ursprünglichen Positionen finden kann.

Über Skripts können diese Dateien komfortabel erzeugt und verwaltet werden.

Bevor aber ausgewählte Dateien hostabhängig gemacht werden können, muss zunächst die Basisstruktur dieser Hierarchie erzeugt werden:

```
com_create_cds1 -n -r /cluster/mount/vg_shared-root-lv_shared-root
/
```

Die Option -n sorgt dafür, dass die Knoten numeriert werden sollen.

Nun muss das Root Verzeichnis auf das gesharete Volume verlegt werden:

```
mount --bind /cluster/mount/vg_shared-root-lv_shared-
root/cluster/cdsl/1 /cluster/mount/vg_shared-root-lv_shared-
root/cdsl.local/
chroot /cluster/mount/vg_shared-root-lv_shared-root/
```

Jetzt kann man beginnen, die hostabhängigen und geshareten Dateien zu erzeugen.:

```
com_create_hostdependent_file -F gfs -n /var
com_create_shared_file -F gfs -n /var/lib
com_create_hostdependent_file -F gfs -n /etc/sysconfig/network
```

Die Option -F gfs sorgt dafür, dass das verwendete Dateisystem nicht automatisch ermittelt wird und die Option -n gibt dem Programm an, dass numerische NodeIDs verwendet werden sollen.

Der Inhalt der Datei */etc/sysconfig/network* muss auf jedem Cluster-Knoten angepasst werden (Hostname, spezielle Netzwerkeinstellungen). In unserem Fall ist das aber trivial, da nur der Hostname zu ändern ist. Einfach kann man diese Änderung über vi vornehmen, da hier mehrere Dateien der cds1-Struktur nacheinander bearbeitet werden können:

```
vi /cluster/cdsl/*/etc/sysconfig/network
```

Als nächstes müssen die statischen Informationen über die einzuhängenden Dateisysteme in der Datei */etc/fstab* angepasst werden.

Hierzu ist es nötig, dass man das Root Dateisystem auf das gesharete Volume verlegt, GFS als Dateisystem einträgt und den Dateisystemcheck beim Systemstart ausschaltet. Ebenfalls ist es eine gute Idee, die Parameter noatime und nodiratime zu setzen, um die Performanz zu erhöhen. Das Boot-Laufwerk sollte nicht automatisch gemountet werden. Eine beispielhafte fstab ist nachfolgend abgedruckt:

```
/dev/vg_shared-root/lv_shared-root /          gfs          defaults      0 0
/dev/sdb1 /boot      ext3         noauto        0 0
none /dev/pts   devpts      gid=5,mode=620 0 0
none /dev/shm  tmpfs       defaults      0 0
none /proc     proc        defaults      0 0
none /sys      sysfs       defaults      0 0
```

Ein weiterer Spezialfall ist dann die Datei */etc/mtab*. In dieser Datei werden die aktuell eingehängten Verzeichnisse verwaltet. Sie muss knotenspezifisch sein. Da allerdings in der Datei */proc/mounts* die gleichen Informationen stehen, kann dieses Problem mit einem symbolischen Link einfach umgangen werden:

```
rm /etc/mtab
ln -s /proc/mounts /etc/mtab
```

Manche Dienste werden von der initialen Ramdisk bereits gestartet und würden mit den defaultmäßig aktivierten Services kollidieren. Andere werden für diesen Workshop einfach nicht benötigt. Deshalb müssen als nächstes die Dienste auf dem Cluster entsprechend angepasst werden: Dies kann entweder manuell geschehen oder man bedient sich folgendem Shell-Skript

```
#!/bin/bash
for SERVICE_OFF in ccsd cman fenced
do
    chkconfig $SERVICE_OFF off
done

for SERVICE_ON in bootsr ccsd-chroot fenced-chroot fenceacksv
do
    chkconfig $SERVICE_ON on
done
```

Nun ist die Sharedroot-Umgebung eingerichtet. Als nächstes ist nun die initiale Ramdisk *initrd* und die bootdisk zu erstellen. Hierzu erstellt man eine bootbare Partition mit mindestens 512MB Speicherplatz, formatiert mit *ext2/3* und mountet das Bootlaufwerk nach */boot*:

```
z.B.
fdisk /dev/sdc --> man fdisk
mkfs.ext3 /dev/sdc1
mount /dev/sdc1 /cluster/mount/vg_shared-root-lv_shared-root/boot
```

Mit *exit* gelangt man aus dem *chroot* heraus und kann dann das Bootlaufwerk duplizieren:

```
copy -ax /boot /cluster/mount/vg_shared-root-lv_shared-root/
```

Anschließend wechselt man wieder in die chroot-Umgebung:

```
chroot /cluster/mount/vg_shared-root-lv_shared-root/
```

Es müssen nun nur noch wenige Modifikationen gemacht werden, dann kann der Diskless Shared Root Cluster booten. Zum Einen muss der Bootloader angepasst werden und zum Anderen muss eine neue initrd erstellt werden.

Falls die Partition völlig neu erstellt wurde, muss Grub erneut installiert werden. Dazu müssen folgende Befehle verwendet werden:

```
grub
> device (hd0) /dev/sdc
> root (hd0,0)
> setup (hd0)
> quit
```

Hierbei wird Grub in den Bootsektor von /dev/sdc installiert. Bitte entsprechend anpassen! Nun muss die Konfigurationsdatei von Grub abgeändert werden, um das Sharedroot zu booten. Eine entsprechende */etc/grub/grub.conf* könnte wie folgt aussehen:

```
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu

initrd_sr-2.6.9-42.0.3.EL.img

title CentOS shared-root (2.6.9-42.0.3.EL)
    root (hd0,0)
    kernel /vmlinuz-2.6.9-42.0.3.EL rw
    initrd /initrd_sr-2.6.9-42.0.3.EL.img

title CentOS (2.6.9-42.0.3.EL)
    root (hd0,0)
    kernel /vmlinuz-2.6.9-42.0.3.EL ro
root=/dev/vg_root/lv_root
    initrd /initrd-2.6.9-42.0.3.EL.img
```

Bitte beachten Sie, dass das initrd der ersten Boot-Stanza auf ein entsprechendes initrd_sr verweist und das komprimierte Kernel-Image vmlinuz-2.6.9-42.0.3.EL nur mit der Option rw angegeben ist. Die Änderungen können sehr leicht durch Kopieren und abändern erreicht werden. Es empfiehlt sich auch, immer die zuletzt funktionierende Konfiguration als Failsafe Eintrag stehen zu lassen!

Als nächstes muss die neue initrd erstellt werden, die in der grub.conf bereits aufgeführt wurde. Dazu ist folgender Befehl zu verwenden:

```
mkinitrd /boot/initrd_sr-$(uname -r).img $(uname -r)
```

Vorraussetzung ist natürlich, dass 2.6.9-42.0.3.EL auch die korrekte Kernelversion ist!

Auf der Konsole sollten Sie folgende Ausgabe sehen können:

```
Extracting rpms...Cannot find rpm "device-mapper-multipath". Skipping.
Cannot find rpm "perl-libxml-perl". Skipping.
Cannot find rpm "perl-Net-SSLeay". Skipping.
[ OK ]
Retreiving dependent files...found 342
[ OK ]
Copying files...
[ OK ]
Copying kernelmodules (2.6.9-42.0.3.EL)...
[ OK ]
builddate_file
[ OK ]
Post settings ..
[ OK ]
Cpio and compress..
[ OK ]
Cleaning up (/tmp/initrd.mnt.mg4812, )...
[ OK ]
-rw-r--r-- 1 root root 39300 Apr 11 10:51 /boot/initrd_sr-2.6.9-42.0.3.EL.img
```

Etwaige "cp: cannot stat" bzw. "Cannot find rpm" Fehler können Sie ignorieren. Es handelt sich hierbei um Abhängigkeiten, die nur für spezielle Weiterentwicklungen und für neueste Features interessant sind.

Die neue initrd wurde nun erstellt und der Cluster ist bereit, gebootet zu werden. Es empfiehlt sich zunächst, immer einen Cluster-Knoten zur Konfiguration zu verwenden und mit dem anderen Cluster-Knoten zu testen. Daher sollte zunächst nur der 2. Knoten gebootet werden, um zu sehen, ob alles richtig konfiguriert wurde.

Sollten sich Fehler eingeschlichen haben, so kann man am verbleibenden Knoten schnell noch Änderungen einpflegen. Wenn kein Knoten mehr im Cluster integriert ist, muss der Node im Rescue Modus gebootet werden. Hier kann dann der Cluster manuell gestartet werden.

Falls dies bei Ihnen nötig sein sollte, die entsprechende Vorgehensweise ist weiter unten beschrieben.

Der Bootprozess sollte ohne Fehler ablaufen und das Login-Prompt sollte erscheinen:

Abschließend muss noch ein kleines Workaround für die aktuelle com.oonics Version eingerichtet werden, damit der Cluster-Knoten sauber herunterfahren kann. In der nächsten Version der com.oonics Pakete wird ein entsprechender Fix enthalten sein.

In der Datei `/etc/init.d/halt` muss folgender Eintrag für `halt_get_remaining()` geändert werden:

```
halt_get_remaining() {
    awk '$2 ~ /^\/$|^\/proc|^\/sys|^\/dev/{next}
        $3 == "tmpfs" || $3 == "proc" {print $2 ; next}
        /^(^#|rootfs|loopfs|autofs|devfs|^none|^\/dev\/ram|^\/dev\/root)/
{next}
        {print $2}' /proc/mounts
}
```

Dailywork mit dem einem Open-Sharedroot Cluster

Der Cluster ist nun konfiguriert und neu gestartet. Jetzt ist es an der Zeit, einige Dinge auszuprobieren, die bei der täglichen Arbeit anfallen können.

Update der Clusterkonfiguration

Im Gegensatz zu herkömmlichen GFS Shared Storage Cluster, muss bei einem Diskless Shared Root Cluster auf Basis von Open-Sharedroot die `Cluster.conf` nicht auf den Knoten als physikalische Kopie verteilt werden. Da das root gesharet wird, sieht jeder Knoten ohnehin die gleiche `Cluster.conf`. Sie muss aber aktiviert werden und in die `initrd` aufgenommen werden:

```
vi /etc/cluster/cluster.conf (Versionsnummer erhöhen!)
ccs_tool update /etc/cluster/cluster.conf
cman_tool version -r <neue Versionsnummer>
cman_tool status (muß die neue Versionsnummer anzeigen)
```

Nun kann eine neue initrd gebaut werden, die die *Cluster.conf* enthält:
`mkinitrd /boot/initrd_sr-$(uname -r).img $(uname -r)`

Manuelles Fencing

Im Regelfall einer normalen Installation wird das Fencing vollautomatisiert ablaufen. Hierzu gibt es hostbasierte Agenten, die einen Node herunterfahren können, falls dieser noch ansprechbar ist, Agenten die über schaltbare Steckdosen einem Clustermitglied den Strom abschalten können oder Agenten, die einem Rechner den Zugang zum gemeinsam genutzten Storage über Zoning entziehen können.

Manuelles Fencing wird normalerweise nur zur Demonstration des Fencings selbst verwendet.

Der folgende Befehl wird auf *host1* oder *host3* ausgeführt und bereitet die Herausnahme von *host2* aus dem Cluster vor.

```
fence_manual -n host2
```

Jetzt muß *host2* manuell aus dem Cluster genommen werden, z.B. durch Abschaltung der entsprechenden virtuellen Maschine.

Der nachfolgende Befehl darf erst auf *host1* oder *host3* ausgeführt werden, nachdem die virtuelle Maschine beendet wurde, bzw. der gefencete Node keinen Zugriff mehr auf die geteilten Ressourcen hat.

```
fence_ack_manual -n host2
```

Der Node *host2* ist nun aus dem Cluster entfernt.

Dailywork: Apache HA Konfiguration

Der Cluster steht nun. Natürlich müssen aber auch noch die Dienste eingerichtet werden, die die Vorteile des Diskless Shared Root Clusters genießen sollen.

Das Beispiel in diesem Workshop zeigt Ihnen, wie eine HA Konfiguration für einen Apache Webserver relativ einfach auf dem Cluster eingerichtet werden kann.

Es handelt sich hierbei um einen einfachen und beliebten Einsatzzweck und es sollte

Ihnen mit dieser Anleitung ohne weiteres gelingen, diese Konfiguration auf einem Testaufbau nachzuvollziehen. In der Praxis ist aber eine Loadbalancing Konfiguration empfehlenswert, die aber schwieriger zu konfigurieren ist.

Im Wesentlichen müssen für die HA-Konfiguration nur zwei Dinge beachtet werden:

1. Das Apache Stop-Script muss zwingend einen Exit-Code von Null generieren, damit der Cluster den Status korrekt überwachen kann.
2. Der *Cluster.conf* muss eine Ressourcen-Gruppe hinzugefügt werden, die alle Knoten enthält, auf denen der Apache laufen soll (Die Cluster-Knoten können unterschiedliche Aufgaben übernehmen!).

Mit den nachfolgend abgedruckten Änderungen an */etc/init.d/httpd* und */etc/cluster/cluster.conf* ist die HA Konfiguration leicht einzurichten.

Die */etc/init.d/httpd* muss mit `exit 0` beendet werden:

```
stop() {
    echo -n $"Stopping $prog: "
    killproc $httpd
    RETVAL=$?
    echo
    [ $RETVAL = 0 ] && rm -f ${lockfile} ${pidfile}
    exit 0
}
```

In der */etc/cluster/cluster.conf* muss folgende Erweiterung eingefügt werden:

```
(...)
<rm>

    <failoverdomains>
        <failoverdomain name="ALL" ordered="0" restricted="1">
            <failoverdomainnode name="host1"
                priority="1"/>
            <failoverdomainnode name="host2"
                priority="1"/>
        </failoverdomain>
    </failoverdomains>
    <resources>
        <ip address="172.16.40.150" monitor_link="yes"/>
        <script file="/etc/init.d/httpd" name="apache"/>
    </resources>
```

```
<service name="web" autostart="1" domain="ALL">
  <ip ref="172.16.40.150"/>
  <script ref="apache"/>
</service>
</rm>
</cluster>
```

Es wird hierbei der Dienst auf Knoten 1 und 2 aktiviert und eine virtuelle IP Adresse 172.16.40.150 zugewiesen. Anfragen, die auf diese Adresse treffen, werden auf dem Cluster lastverteilt. Der Apache Webserver kann ganz regulär konfiguriert werden. Die Knoten müssen nicht über identische Hardware verfügen, bzw. es können auch auf den Knoten mehrere Anwendungen (z.B. httpd, mysqld, postfix usw.) simultan laufen. Mit der priority Einstellung kann die Leistungsfähigkeit und die verbleibenden Ressourcen der Cluster-Knoten berücksichtigt werden, wenn im Fehlerfall die Anwendung auf die verbleibenden Knoten verschoben wird.

Dailywork: Statisches NFS Loadbalancing

Ein Cluster mit mit geshartem Dateisystem eignet sich für viele Anwendungen sehr gut für Loadbalancing. Anhand von NFS soll gezeigt werden, wie einfach man eine statische Lastverteilung auf einem Sharedroot Cluster einrichten kann.

Hier sind keine weiteren Anpassungen an der */etc/cluster.conf* notwendig. Das statische NFS Loadbalancing kann also parallel zur Apache HA Konfiguration implementiert werden.

Dazu muss auf dem Diskless Shared Root Cluster ein Verzeichnis als NFS exportiert werden. Hierzu trägt man in der Datei */etc/exports* folgendes ein:

```
/var/nfs/pub          172.16.40.1(rw)
```

Dieses Verzeichnis erstellt man, setzt die Rechte und befüllt es mit Testdaten:

```
mkdir -p /var/nfs/pub
chmod og+rwx /var/nfs/pub
touch /var/nfs/pub/testdaten.dat
```

Nun muss noch NFS aktiviert werden:

```
chkconfig nfs on
service nfs start
```

```
exportfs
```

Nun kann von der IP 172.16.40.1 (ebenfalls der vmware-host) auf das Share über unterschiedliche Cluster-Knoten zugegriffen werden:

```
mount -t nfs host1:/var/nfs/pub /mnt/temp
```

bzw.:

```
mount -t nfs host2:/var/nfs/pub /mnt/temp
```

Jeder Cluster-Knoten exportiert das gleiche Verzeichnis. Dadurch ist ein statisches Loadbalancing möglich. Ein IP-Range kann beispielsweise über host1 darauf zugreifen, ein anderer IP-Range über host2. Man könnte die Clients mittels DNS auch über ein RoundRobin Verfahren auf das exportierte Dateisystem lenken.

Da bei NFS Schreibzugriffe den limitierenden Faktor darstellen, ist eine Konfiguration mit statischem Loadbalancing vernünftig.