Clustern mit Xen

Xen gilt mittlerweile als Standardlösung für die Virtualisierung unter Linux. Es kann aber noch mehr als nur Server konsolidieren. Auch Cluster, bei denen im Fehlerfall ein Service mit seiner virtuellen Maschine einfach auf einen anderen Host migriert, lassen sich leicht aufsetzen. Sie können viele Dienste mit geringem Hardwareaufwand effektiv sichern. Ein Beispiel auf der Grundlage von RHEL 5 demonstriert dieser Beitrag. Thorsten Scherf

Das hier vorgestellte Beispiel arbeitet mit einem mehrstufigen Konzept für die Ausfallsicherheit : Gleich zwei Cluster sind im Spiel (**Abbildung 1**). Die untere Ebene bildet ein Cluster-Pärchen aus Xen-Servern; sie schützt die Dom0-Hosts, die hier Node1 und Node2 heißen sollen und beide unter Red Hat Enterprise Linux 5 (RHEL5) laufen. Wenn einer dieser Knoten ausfällt, kann der zweite mit dem Shared Storage weiterarbeiten, einem iSCSI-Spiegel mit Cluster-Dateisystem, auf den beide zugreifen können. Auf diese Weise ist immer wenigstens eine Xen-Instanz erreichbar und aktiv.

Jede Xen-Instanz startet im Beispiel eine virtuelle Maschine (VM1 und VM2), die ebenfalls mit RHEL5 realisiert sind. Jede dieser virtuellen Maschinen bietet einen Dienst an, hier ist es ein Webserver. Der Dienst wird durch einen zweiten DomU-Cluster überwacht, der als Reaktion auf einen Ausfall den Webserver innerhalb der zweiten virtuellen Maschine auf dem anderen DomU-Cluster-Knoten neu startet.

Auf noch mehr virtuelle Maschinen verzichtet die Modellinstallation bewusst, um das Setup nicht zu komplex zu gestalten. Im produktiven Einsatz lassen sich natürlich weitere Xen-Gäste pro Node installieren, sofern entsprechend leistungsfähige Hardware für die Hosts zur Verfügung steht.

Ausfallsicherer Webserver

Den Webserver Apache auf VM1 richtet der Administrator mithilfe der Red Hat Cluster Suite ein und ordnet ihm eine virtuelle IP (VIP) zu. Sollte dieser Dienst nicht mehr erreichbar sein, migriert der Service automatisch auf die andere virtuelle Maschine VM2 und startet dort. Wie schon beschrieben, läuft die zweite Xen-Instanz auf dem anderen physikalischen Node. Somit ist auch im Falle eines Hardwarefehlers von Node1 sichergestellt, dass der Webserver weiter benutzbar bleibt. Die virtuelle IP des Webservers und das verwendete Filesystem mit den Webserverdaten wandern im Zuge des Failover ebenfalls zum zweiten Knoten des Xen-Cluster. Für den Benutzer des Webservers ist dieser Vorgang vollkommen transparent. Er bemerkt den Ausfall gar nicht.

Den Shared Storage stellt in diesem Beispiel ein iSCSI-Server bereit, eine kostengünstige Alternative zu Fibre-Channel-SAN-Lösungen. Bei heutigen Netzwerkbandbreiten bedeutet das keinen großen Performanceverlust mehr.

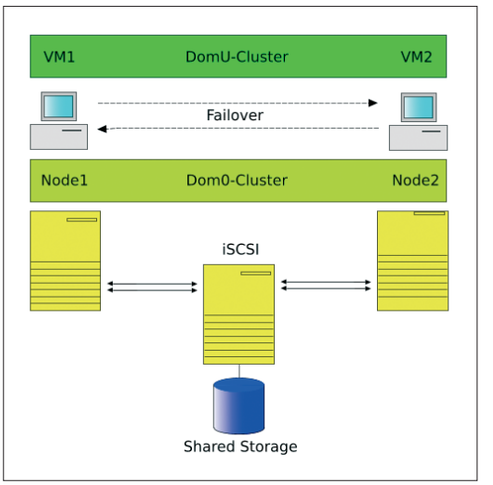

Abbildung 1: Nach diesem Architekturschema arbeitet der zweistufige Cluster für virtuelle Maschinen, die beliebige Dienste ausfallsicher anzubieten vermögen.

Die hier verwendete Distribution bringt bereits einen iSCSI-Client mit, aber keinen Server. Den kann man einfach von (1) als RPM-Paket herunterladen und installieren. Alternativ kommt natürlich auch eine Hardwarelösung (etwa der AX100-Server von Clariion oder das Provigo 410 von Transtec) in Frage.

Damit die Netzwerkverbindung zum iSCSI-Server keinen Single Point of Failure bildet, arbeitet die Musterlösung mit Multipathing: Cluster und iSCSI-Server verbinden also zwei voneinander unabhängige Netzwerkpfade. Sollte ein Anschluss beispielsweise durch einen Fehler der Netzwerkkarte oder des Switches ausfallen, bleibt der exportierte RAID-1-Spiegel des iSCSI-Servers immer noch über das zweite Netzwerk erreichbar. Eine dritte Netzwerkkarte der beiden Nodes vermittelt über die virtuelle IP-Adresse den Kontakt zum Webservice. Diese Karte ist über eine Bridge mit den Xen-Instanzen des jeweiligen Knotens verbunden.

Teilen sich Cluster-Knoten eine Ressource, darf in der Regel zu einer Zeit dennoch immer nur einer exklusiv das Zugriffsrecht erhalten. Versuchten etwa mehrere Rechner, ein gewöhnliches Filesystem zu mounten und gleichzeitig darauf zu schreiben, wäre Datenverlust nur eine Frage der Zeit. (Die Ausnahme wäre hier ein Cluster-fähiges Filesystem, das parallele Zugriffe erlaubt.) Solange also ein Knoten die Ressource beansprucht, gilt es, andere davon auszuschließen. Man spricht von Fencing (Abzäunen). ▶

Dafür stehen diverse Methoden zur Verfügung. Im einfachsten Fall lassen sich die lokalen Management-Boards der Nodes (iLO, DRAC und so weiter) verwenden, um den auszuschließenden Knoten stillzulegen. Diese Variante wird auch mit der Abkürzung STONITH belegt (Shoot The Other Node In The Head). Als so genanntes Fencing Device lassen sich aber auch via Netzwerk schaltbare Power Switches, die kurzzeitig die Stromversorgung für einen Node unterbrechen, oder SAN-Switches verwenden, die den entsprechenden SAN-Port sperren.

In dieser Konfiguration ist es der DomU-Cluster, der sicherstellen muss, dass nur eine virtuelle Maschine auf das Filesystem des Webservers zugreift. Um die andere daran zu hindern, existiert in der Red Hat Cluster Suite speziell für virtuelle Maschinen ein Fence Agent, »fence_xvm«. Im Dom0-Cluster ist der Daemon »fence_xvmd« zu starten, der die einzelnen Xen-Maschinen über den »fence_xvm«-Agent überwacht.

Reagiert eine virtuelle Maschine nicht mehr, beendet sie »fence_xvmd« und startet sie neu. Der Webserver ist in der Zwischenzeit bereits auf die zweite virtuelle Maschine umgezogen.

Der iSCSI-Server

Der iSCSI-Server erhält als erstes einen Plattenspiegel (Software-RAID-1), um vor Datenverlust nach Festplattenausfall geschützt zu sein. (Wer einen Hardware-RAID-Controller hat, benutzt natürlich den.) RHEL5 verwendet dafür seit einiger Zeit die Software »mdadm«. Vorausgesetzt, die nötigen Partitionen existieren bereits, lässt sich der Spiegel mit dem folgenden Kommando einrichten:

```
[root@iscsi ~]# mdadm -C /dev/md0 -l 0 ➥
-n 2 /dev/sdb1 /dev/sdc1
```

Das so erzeugte Device »/dev/md0« kann nun die iSCSI-Target-Software den beiden Nodes über das Ethernet zur Verfügung stellen. Hier-

Cluster Suite

Die Infrastruktur der Red Hat Cluster Suite besteht besteht aus mehreren Komponenten: dem Cluster-Manager, dem Lock-Manager, dem Fence Daemon und dem Cluster Configuration System.

Als Cluster-Manager kommt unter RHEL5 der CMAN (Cluster-Manager) zum Einsatz. Dieser kümmert sich darum, einzelne Nodes zum Cluster hinzuzufügen oder wieder zu entfernen und im Falle eines Node-Ausfalls entsprechende Maßnahmen einzuleiten. Der Cluster-Manager entscheidet, auf welchen Cluster-Rechnern ein Dienst gestartet werden darf und auf welchen nicht.

Als Lock-Manager setzt RHEL5 heute ausschließlich den DLM (Distributed Lock Manager) auf allen Knoten ein. Andere Cluster-Ressourcen wie das Global File System können für das Locking von Dateien auf den DLM zurückgreifen. Für bestimmte Setups (etwa für Oracle RAC) wird noch der ältere GULM Lock-Manager benötigt, der externe Lock-Server zur Verfügung stellt.

Das Fence-System besteht aus einem Fence Daemon, einem Fence Agent und einem Fence Device. Der Fence Agent wird vom Cluster-Manager (CMAN) über ausbleibende Statusmeldungen eines anderen Cluster-Knotens informiert. Der Fence Daemon versucht dann über den Fence Agent und dem damit verknüpften Fence Device, den ausgefallenen Node zu rebooten. Als Fence Devices kommen üblicherweise Power Switches oder auch SAN-Switches zum Einsatz. Bei Letzteren wird im Bedarfsfall einfach der Zugang zum Storage verhindert, indem der Switch Port des Host offline geschaltet wird.

Das Cluster Configuration System (CCS) besteht aus einem Daemon, der sich exklusiv um die Konfiguration des Cluster kümmert. Nach jeder Änderung der Konfigurationsdatei »/etc/cluster/cluster.conf« benachrichtigt er eigenverantwortlich die anderen Nodes und versorgt sie mit einer neuen Kopie der Konfiguration.

Die einzelnen Dienste dieser Core-Infrakstruktur startet das Init-Skript »/etc/init.d/cman«. Neben dieser Core-Infrastruktur gibt es darauf aufbauend weitere Cluster-Komponenten: Das Service-Management, das Cluster Logical Volume Management und das Cluster-Dateisystem GFS.

Für hochverfügbare Cluster-Dienste ist der »rgmanager« zuständig. Er startet den Cluster-Dienst beim Ausfall einer Machine auf einer anderen Maschine des Cluster neu. Zu seinen Aufgaben zählt auch das Einbinden des benötigten Dateisystems sowie die Aktivierung der virtuellen IP-Adresse, die zu dem Cluster-Dienst gehört.

Über den CLVMD (Cluster Logical Volume Manager) lassen sich Volume Groups und deren Logical Volumes im Cluster betreiben. Die hierfür verwendeten Metainformationen stehen eigentlich nur lokal auf einer Maschine zur Verfügung, werden aber durch den CLVMD zwischen den einzelnen Cluster-Knoten synchronisiert.

GFS ist das Global File System, das einen gleichzeitigen Zugriff von mehreren Nodes auf ein Blockdevice ermöglicht. Es ist ein natives Filesystem und baut direkt auf dem VFS (Virtual File System) des Linux-Kernel auf. Durch den Einsatz von mehreren Journals und durch das Synchronisieren der Metadaten ist es bestens als Cluster-Dateisystem geeignet.

für sind nur wenige Einträge in der Datei »/etc/inetd.conf« notwendig:

```
Target iqn.2007-05.com.example:storage.➥
iscsi.disk1
Lun 0 Path=/dev/md0,Type=fileio
```

Das Init-Skript »/etc/init.d/iscsi-target« startet anschließend den Server.

Cluster-Verwaltung mit Conga

Für die Verwaltung der beiden Cluster Dom0 (Node1/Node2) und DomU (VM1/VM2) kommt das Tool Conga zum Einsatz, das hier ebenfalls auf dem iSCSI-Server läuft. Natürlich könnte man es auch auf einem anderen Rechner starten. Das notwendige RPM-Paket »luci« befindet sich im Cluster-Repository der RHEL5-Distribution und kann beispielsweise über das Red Hat Network bezogen werden. Nachdem das Tool eingerichtet ist, lässt es sich mittels »service luci start« aktivieren. Die Konfiguration übernimmt der Befehl »luci_admin init«. Hierbei wird unter anderem das Zertifikat erzeugt, das für eine gesicherte HTTPS-Verbindung zum Conga-Webinterface (**Abbildung 3**) notwendig ist. Für den ersten Zugriff auf Conga über dieses Webinterface benötigt man später das in diesem Schritt vergebene Passwort. Eine ausführliche Anleitung zu Conga findet sich unter **(2)**.

Beide physischen Nodes arbeiten mit dem jeweils aktuellsten RHEL5 Xen-Kernel. Dieser und alle weiteren Xen-Tools lassen sich aus dem RHN bequem mittels »yum groupinstall Virtualization« aufspielen.

Die Nodes benötigen noch etwas Handarbeit für die Einrichtung. Das fängt beim Setup des Dom0-Cluster an, der für das Fencing der beiden Nodes zuständig ist, sollte einer ausfallen. Dafür installiert der Admin die Red Hat Cluster Suite 5, die unter anderem das Paket »ricci« mitbringt, das mit dem zentralen Managementtool Conga – genauer der dortigen Komponente Luci – kommuniziert. Ist die Maschine im Red Hat Network registriert, geschieht die Installation der Cluster-Software einfach mittels »yum groupinstall Cluster«.

Über das Webinterface von Conga lässt sich nun der erste Cluster einrichten. Dafür ist die URL aufzurufen, die bei der Installation von Luci auf dem iSCSI-Server ausgegeben wurde. Im vorliegenden Fall ist das **[https://iscsi.example.com:8084/luci/homebase]**. Nach einem erfolgreichen Login sollte man als Erstes die beiden

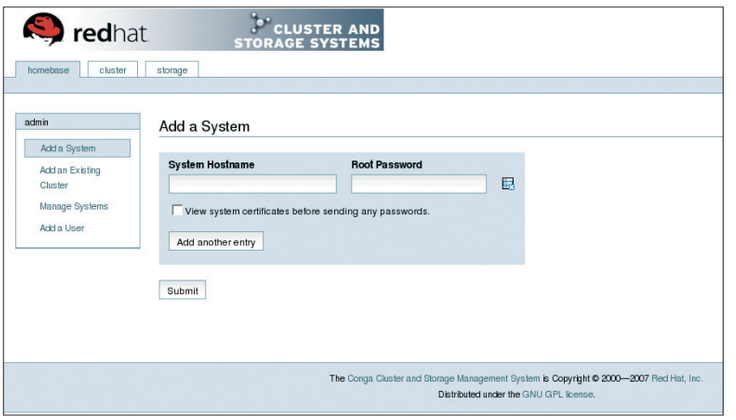

Abbildung 2: Über das Webinterface von Conga lässt sich auf dieser Seite ein neuer Node dem Cluster hinzufügen.

physischen Nodes über »Add a System« hinzufügen (**Abbildung 2**). Über den »Cluster«-Tab und »Create a new Cluster« lässt sich dann der eigentliche Dom0-Cluster einrichten. Er wird Dom0 genannt, und seine Mitglieder sind Node1 und Node2.

Die Software richtet jetzt die beiden Knoten automatisch ein und startet die notwendigen Dienste. Hat alles geklappt, erscheint nach einiger Zeit eine neue Seite für die weitere Cluster-Konfiguration. Auf ihr ist unbedingt die Checkbox »Run XVM fence daemon« zu aktivieren, damit der entspreche Daemon »fence_xvmd« im Dom0-Cluster automatisch startet.

Ein Klick auf die einzelnen Nodes des Cluster eröffnet die Möglichkeit, Fence Agents für diese Nodes auszuwählen, für HP-Blades also beispielsweise den Agent »HP iLO«, der in der schon beschriebenen Weise das Management-Board der Blades verwendet, um einen ausgefallenen Node zu rebooten. Die Funktionsweise ähnelt der eines Power Switches, nur dass hier der Power Switch lokal in Form des Management-Device vorhanden ist.

iSCSI einbinden

Die erzeugte Datei sollte nun in etwa so aussehen wie in **Listing 1** dargestellt. Die Konfiguration überträgt Conga automatisch auch auf den anderen Cluster-Knoten. Für eine genauere Beschreibung der möglichen Konfigurationseinstellungen von Conga empfiehlt sich das Studium der ausführlichen Anleitung **(3)**.

Der nächste Schritt bindet den exportierten iSCSI-Spiegel auf den beiden Nodes ein. Dafür ist auf beiden Hosts das Paket »iscsi-initia-

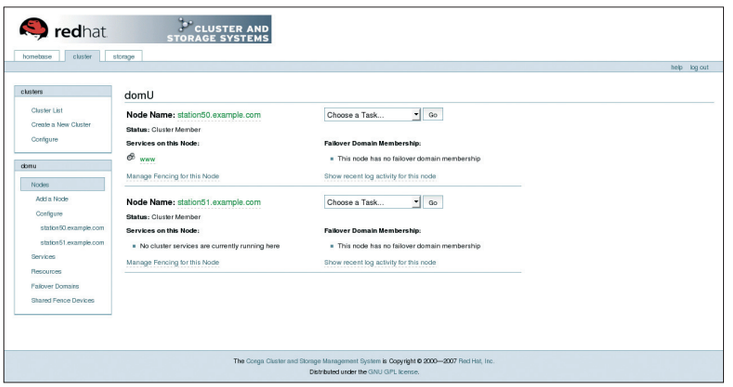

Abbildung 3: Der DomU-Cluster in der Darstellung des Tools Conga. Er beherbergt den Webservice, der hochverfügbar ausgelegt werden soll.

tor-utils« nötig. Es ist Bestandteil der RHEL5-Distribution und ist einfach mittels »yum« aus dem Red Hat Network zu beziehen. Die folgenden Kommandos zeigen die zur Verfügung stehenden Exporte des iSCSI-Servers an. So lassen sich diese schließlich einbinden:

```
[root@station5 ~]# iscsiadm -m discovery➥
-t st -p 172.17.1.1
172.17.1.1:3260,1 iqn.2007-05.com.example:➥
storage.iscsi.disk1

[root@station5 ~]# iscsiadm -m discovery➥
-t st -p 172.17.2.1
172.17.1.1:3260,1 iqn.2007-05.com.example:➥
storage.iscsi.disk1
172.17.2.1:3260,1 iqn.2007-05.com.example:➥
storage.iscsu.disk1

[root@station5 ~]# iscsiadm -m node -T
iqn.2007-05.com.example:storage.iscsi.disk1➥
-p 172.17.1.1
[root@station5 ~]# iscsiadm -m node -T
iqn.2007-05.com.example:storage.iscsi.disk1➥
-p 172.17.2.1
```

Danach sollte auf dem ersten Node beim Aufruf von »fdisk -l « nicht nur die lokale Festplatte »/dev/sda« zu sehen sein, sondern auch der exportierte iSCSI-Spiegel. Weil es zwei Netzwerkpfade zum Server gibt, taucht das exportierte Device jetzt doppelt auf.

Der gestartete Dienst »iscid« sorgt dafür, dass die auf diese Weise erzeugten Devices auch nach einem Reboot wieder zur Verfügung stehen. Die hierfür notwendigen Informationen liegen bereits nach dem allerersten Discovery im Ordner »/var/lib/iscsi«.

Multipathing

Um beide Pfade zum Spiegel über eine gemeinsames Device-Datei nutzen zu können, muss man die beiden Devices »/dev/sdb« und »/dev/sdc« zu einem Multipath Device zusammenschalten. Das bewerkstelligt das Paket »device-mapper-multipath«, das ebenfalls bereits zur RHEL5-Distribution gehört. In seiner Konfigurationsdatei »/etc/multipath.conf« sind per Default alle Devices in einer Blacklist verzeichnet, der Multipath Daemon beachtet sie deshalb nicht. Um ihn auf das neue Device aufmerksam zu machen, kommentiert man es in der Blacklist wie folgt aus:

```
#blacklist {
#       devnode "*"
#}
```

Der Start des Multipath Daemon mittels »service multipathd start« erzeugt dann das gewünschte Multipath Device unter »/dev/mapper/«. Die Struktur des Spiegels lässt sich nun ganz einfach mit dem folgenden Kommando ausgeben:

```
[root@station5 ~]# multipath -l
mpath1 (149455400000000000000000001000000➥
6919000010000000) dm-2 IET,VIRTUAL-DISK
[size=4.7G][features=0][hwhandler=0]
\_ round-robin 0 [prio=0][active]
 \_ 2:0:0:0 sdc 8:32  [active][undef]
\_ round-robin 0 [prio=0][enabled]
 \_ 1:0:0:0 sdb 8:16  [active][undef]
```

Es sind die beiden Pfade »/dev/sdb« (via 172.17.1.1) und »/dev/sdc« (via 172.17.2.1) zu erkennen. Beide Geräte (die der iSCSI-Server ja zu dem RAID-1-Device »/dev/md0« zusammenfasst) lassen sich nun einheitlich über die Device-Datei »/dev/mapper/mpath1« ansprechen. Mit Hilfe des Kommandos »fdisk /dev/mapper/mpath1« lassen sich nun einfach auf diesem Spiegel die gewünschten Partitionen einrichten.

Zunächst benötigt der Admin hier eine Partition, die die Xen-Images für die beiden Xen-Instanzen VM1 und VM2 aufnimmt. Eine zweite Partition kann man in diesem Schritt ebenfalls gleich anlegen. Sie dient später als Document-Root des hoch verfügbaren Webservers und wird als Ressource im DomU-Cluster definiert. Sind die Partitionen eingerichtet, mappt sie das Tool »kpartx« ins Verzeichnis »/dev/mapper«. Nun lassen sie sich im nächsten Schritt mit einem Filesystem versehen:

```
[root@station5 ~]# kpartx -l➥
/dev/mapper/mpath1
mpath1p1 : 0 8803557 /dev/mapper/mpath1➥
63
mpath1p2 : 0 963900 /dev/mapper/mpath1 ➥
8803620

[root@station5 ~]# kpartx -a ➥
/dev/mapper/mpath1
```

Formatieren mit GFS

Hat dies geklappt, lässt sich die erste Partition mit dem Cluster-Dateisystem GFS formatieren. Durch den Einsatz eines Cluster-fähigen Dateisystems für den Shared Storage zwischen den beiden Dom0-Nodes ist sichergestellt, dass dieses bei einem parallelen Zugriff nicht beschädigt wird. Die notwendige Software findet sich im RHN-Channel RHEL5-ClusterStorage und lässt sich einfach mittels »yum groupinstall ClusterStorage« installieren. Der folgende Befehl bringt das Filesystem auf das soeben eingerichtete Multipath Device:

```
[root@station5 ~]# gfs_mkfs -p lock_dlm ➥
-t Dom0:gfs1 -j 2 /dev/mapper/mpath1p1
```

Als Lock-Manager kommt unter RHEL5 der DLM (Distributed Lock Manager) zum Einsatz. Die Option »-j« gibt die Anzahl der zu verwendenden Journale an. Diese Anzahl ist identisch mit der Anzahl der Nodes, in unserem Fall also zwei. Abschließend trägt man die Partition in die Datei »/etc/fstab« ein und mountet sie. Der Aufruf von »df« sollte nun die neue Partition anzeigen:

```
[root@station5 ~]# df -h
Filesystem   Size  Used Avail Use%
Mounted
/dev/mapper/vol0-root➥
             7.8G  2.4G  5.1G  32% /
/dev/sda1     99M   35M   60M  37% /boot
tmpfs        948M     0  948M   0% /dev/
shm
/dev/mapper/vol0-home➥
             496M   19M  452M   4% /home
/dev/mapper/mpath1p1➥
             4.7G   67M  4.6G   2% /Xen
```

Auf dem zweiten Node ist der exportierte iSCSI-Spiegel ebenfalls einzubinden. Die Multipath-Konfiguration lässt sich von Node1 übernehmen. Das Device »/dev/mapper/mpath1p1« sollte nach dem Start des Daemon sofort zu sehen sein und ist per Eintrag in »/etc/fstab/« mit anschließendem »mount -a« zu mounten.
Damit ist das Grundgerüst fertig, und es kann, nach einem großen Schluck aus der Kaffeetasse, an das Einrichten der Xen-Gäste und des zweiten Cluster (DomU) gehen. ▶

Listing 1: Cluster-Konfiguration Dom0

```
01 <?xml version="1.0"?>
02 <cluster alias="dom0" config_version="10"
    name="dom0">
03   <fence_daemon clean_start="0" post_fail_
      delay="0" post_join_delay="12"/>
04     <clusternodes>
05       <clusternode name="station6.example.com"
          nodeid="1" votes="1">
06         <fence>
07           <method name="1">
08             <device name="ilo2"/>
09           </method>
10         </fence>
11       </clusternode>
12      <clusternode name="station5.example.com"
         nodeid="2" votes="1">
13          <fence>
14            <method name="1">
15              <device name="ilo"/>
16            </method>
17          </fence>
18       </clusternode>
19     </clusternodes>
20    <cman expected_votes="1" two_node="1"/>
21   <fencedevices>
22   <fencedevice agent="fence_ilo"
   hostname="172.17.2.100" login="user" name="ilo"
   passwd="pass"/>
23   <fencedevice agent="fence_ilo"
   hostname="172.17.2.110" login="user" name="ilo2"
   passwd="pass"/>
24 </fencedevices>
25 <rm>
26   <failoverdomains/>
27   <resources/>
28 </rm>
29 <totem consensus="4800" join="60" token="10000"
   token_retransmits_before_loss_const="20"/>
30 <fence_xvmd/>
31 </cluster>
```

Die Gäste

Wer bereits über RHEL5-Xen-Images verfügt, der kopiert sie einfach von einem Node aus in das Verzeichnis »/Xen«. Physisch liegen die Images somit auf dem RAID-1-Spiegel des iSCSI-Servers. Muss man die Gastmaschinen neu installieren, bietet sich dafür das Kommandozeilentool »virt-install« oder das grafische Frontend »virt-manager« an. Unter (7) findet sich eine genaue Anleitung, wie diese Installation von Xen-Maschinen funktioniert.

Wichtig ist, dass die erzeugte Image-Datei auf dem Multipath Device »/dev/mapper/mpath1p1« landet, das auf den Nodes unter dem Mountpoint »/Xen« eingehängt wurde.

Bei der Installation der Gäste installiert man direkt die notwendige Cluster-Software. Existiert sie noch nicht, kann sie der Admin nun aufspielen, wie im **Kasten „Software-Setup“** beschrieben. Da die Gastmaschinen auf eine zweite Partition zugreifen müssen, sobald der Webserver auf einer Maschine gestartet wurde, ist die Konfigurationsdatei der Xen-Gäste nachträglich, wie im **Listing 1** dargestellt, anzupassen. Die Partition »/dev/xvdb« wird anschließend als Cluster-Ressource im DomU-Cluster eingetragen und als DocumentRoot vom Webserver verwendet.

Sind die virtuellen Xen-Instanzen soweit vorbereitet und gestartet, kann man sie dem Conga Managementtool bekannt machen. Im oben beschriebenen Webinterface von Conga ist dafür der Menüpunkt »Add a System« zuständig, der die beiden virtuellen Maschinen VM1 und VM2 integriert. Der DomU-Cluster lässt sich nun über den Punkt »create« einrichten.

Der Unterschied zum Dom0-Cluster besteht darin, das als Fence Device für die virtuellen Maschinen nun »Virtual Machine Fencing« auszuwählen ist. Hierdurch wird der der »fence_xvm«-Agent gestartet, der mit dem »fence_xvmd«-Daemon aus dem Dom0-Cluster kommuniziert und im Zweifelsfall eine virtuelle Maschine rebootet.

Software-Setup

Die eingesetzte Software lässt sich mit dem Tool »yum« direkt aus dem RHN laden und installieren, solange die jeweilige Maschine dort regis-triert und mit den entsprechenden Softwarekanälen verknüpft ist. Alternativ taugen auch die Installations-CDs für das Aufspielen.

Anstelle von RHEL5 ist auch Fedora benutzbar. Die Software wird von »yum« dann nicht aus dem Red Hat Network, sondern aus den entsprechenden Fedora-Repositories bezogen. Allerdings steht für eine Cluster-Installation unter Fedora das Conga-Setup-Tool noch nicht zur Verfügung, sodass Sie stattdessen auf das ältere Tool »system-config-cluster« zurückgreifen müssen. Eine nähere Beschreibung zu dem Tool finden Sie unter (6). Im Beispiel wurde folgende Software eingesetzt:

```
iSCSI-Server:
RHEL5-Server
Cluster-Verwaltungstool »luci« (Conga)
iSCSI Target (http://iscsitarget.sourceforge.net/)
Alternativ: Hardwarelösung, zum Beispiel Clariion AX100

Node1/Node2:
RHEL5-Server (RHN-Channel: RHEL5-Server)
Red Hat Cluster Suite (RHN-Channel: RHEL5-Cluster)
Global File System (GFS) (RHN-Channel: RHEL5-ClusterStorage)
iscsi-utils (RHN-Channel: RHEL5-Server)
device-mapper-multipath (RHN-Channel: RHEL5-Server)

VM1/VM2:
RHEL5-Server (RHN-Channel: RHEL5-Server)
Red Hat Cluster Suite (RHN-Channel: RHEL5-Cluster)
httpd (RHN-Channel: RHEL5-Server)
```

Schlüsselfrage

Für eine authentifizierte Kommunikation zwischen dem »fence_xvmdc« und dem »fence_xvm« in den einzelnen Gastmaschinen ist ein Key notwendig, der momentan noch manuell zu erzeugen und auf alle Cluster-Nodes (auch auf die virtuellen) zu kopieren ist. Der Key lässt sich leicht mit folgendem Befehl erzeugen:

```
dd if=/dev/urandom of=/etc/cluster/➥
fence_xvm.key bs=4096 count=1
```

Danach verteilt »scp« den Key auf die einzelnen Nodes. Wichtig ist, dass beim Kopieren unbedingt der Dateiname »/etc/cluster/fence_xvm« erhalten bleibt.

Ein weiterer Unterschied zum Dom0-Cluster besteht darin, dass im DomU-Cluster nun der eigentliche hochverfügbare Dienst in Form eines Webservers ins Spiel kommt, der bei einem Maschinenausfall auf einer anderen Xen-Instanz neu starten soll.

Über den Menüpunkt »Cluster List« auf der linken Seite des Webinterfaces sollten nun beide Cluster, also Dom0 und DomU, zu sehen sein. Ein Klick auf den Dom0-Cluster bringt die beiden Nodes Node1 und Node2 zum Vorschein, die sich über einen Klick auf ihren Namen konfigurieren lassen. Das gleiche gilt für die Nodes des Clusters DomU, die ein Klick auf den zweiten Cluster anzeigt.

Nun ist es an der Zeit, den eigentlichen Webserver einzurichten. Nachdem der DomU-Cluster ausgewählt wurde, lässt sich über den Punkt »Services« und »Add a Service« der Webserver zum Cluster hinzufügen. Ist der Dienst benannt, sind ihm verschiedene Ressourcen zuzuordnen. Hierzu zählen das Init-Skript des Webservers, das Filesystem, das von der jeweils aktiven Maschine eingebunden wird und dem Webserver als DocumentRoot zur Verfügung steht, sowie eine feste IP-Adresse, unter der der Dienst erreichbar ist.

Speicher für Webserver

Für das Filesystem kommt das in der Xen-Konfiguration angegebene Shared Storage zum Einsatz. Dieses Device steht innerhalb der Xen-Maschinen als »/dev/xvdb1« zur Verfügung. Die jeweils aktive Machine bindet es ein. Als Filesystem kann man hier Ext3 verwenden, da es keine konkurrierende Zugriffe von mehreren Maschinen gibt. Die IP-Adresse des Dienstes beansprucht ebenfalls immer die aktive Maschine und bindet sie als zweite Adresse an die Netzwerkkarte »eth0«. In gezeigten Beispiel ist das die Adresse 192.168.0.60. Der Dienst ließe sich nun noch einer so genannte Failover-Domain zuordnen. Dies ist in diesem Lab Setup allerdings nicht sinnvoll. Ganz anders lägen die Dinge, wären mehrere Maschinen mit mehreren Services vorhanden. Dann legt die Failover-Domain innerhalb des Cluster fest, auf welchen Gastmaschinen ein Service gestartet werden darf. Damit wird verhindert, dass im Zuge eines

Listing 2: »/etc/Xen/vm1«

```
01 # Automatically generated Xen config file
02 name = "vm1"
03 memory = "512
04 disk = [ 'tap:aio:/Xen/vm1.img,xvda,w', 'phy:mapper/
   mpath1p2,xvdb1,w',]
05 vif = [ 'mac=00:16:3e:71:a5:8b, bridge=Xenbr0', ]
06 vnc=0
07 vncunused=1
08 uuid = "154cde6d-42b0-c2b7-7f52-bfaa2f44c1a9"
09 bootloader="/usr/bin/pygrub"
10 vcpus=1
11 on_reboot   = 'restart'
12 on_crash    = 'restart'
```

Listing 3: »Cluster-conf.domU«

```
01 <?xml version="1.0"?>
02 <cluster alias="domU_cluster" config_version="4"
    name="domU">
03   <fence_daemon post_fail_delay="0" post_join_
     delay="3"/>
04     <clusternodes>
05       <clusternode name="station50.example.com"
          nodeid="1" votes="1">
06         <fence>
07           <method name="1">
08             <device domain="cluster1"
                name="virt_fence"/>
09           </method>
10         </fence>
11       </clusternode>
12       <clusternode name="station51.example.com"
          nodeid="2" votes="1">
13         <fence>
14           <method name="1">
15             <device domain="cluster2"
                name="virt_fence"/>
16           </method>
17         </fence>
18       </clusternode>
19     </clusternodes>
20     <cman expected_votes="1" two_node="1"/>
21       <fencedevices>
22         <fencedevice agent="fence_xvm"
            name="virt_fence"/>
23       </fencedevices>
24      <rm>
25      <resources>
26          <script file="/etc/rc.d/init.d/httpd"
             name="httpd"/>
27          <fs device="/dev/xvdb2" force_fsck="0"
              force_unmount="1" fsid="6443"
28            fstype="ext3" mountpoint="/var/www/
              html/" name="httpd-content"
              options="" self_fence="1"/>
29           <ip address="192.168.0.60" monitor_
              link="1"/>
30 31      </resources>
32         <service autostart="1" domain="httpd"
            name="httpd">
33         <ip ref="192.168.0.60"/>
34         <fs ref="httpd-content"/>
35       <script ref="httpd"/>
36     </service>
37   </rm>
38 </cluster>
```

Failover alle Services ungewollt auf einer einzelnen Maschine landen. Das Hinzufügen einer solchen Failover-Domain ist allerdings nicht zwingend nötig.

Abschlusstest

Nachdem damit die notwendigen Services angelaufen sind, ist der Cluster-Dienst jetzt fertig eingerichtet. Im hier vorgestellten Beispiel ist er jetzt unter seiner virtuellen IP-Adresse 192.168.0.60 oder auch alternativ über die Beispiel-URL **[http://www.example.com]** für den Anwender benutzbar.

Netzwerk-Setup

Das Beispiel verwendet drei Netze, von denen zwei (172.17.1.0/24 und 172.17.2.0/24) für die redundante Verbindung zum iSCSI-Server dienen. Das dritte Netz (192.168.0.0/24) wird sowohl für die Cluster-Kommunikation als auch für den Zugriff auf den Webserver gebraucht. Es ist natürlich möglich, die Cluster-Kommunikation über ein eigenes Netz zu reservieren, worauf das hier vorgestellte Setup verzichtet.

Die virtuelle IP 192.168.0.60 (www.example.com), die der hoch verfügbare Webserver verwendet, wird jeweils als zweite IP-Adresse an die Schnittstelle »eth0« der virtuellen Maschine gebunden, auf der der Dienst läuft. Beim Setup der Maschinen ist darauf zu achten, dass alle verwendeten Rechnernamen über einen DNS-Server (oder noch besser über eine synchronisierte Datei »/etc/hosts«) von allen beteiligten Rechnern auflösbar sind. Synchronisiert der Admin die lokalen Rechneruhren mittels NTP (Network Time Protocol), hilft ihm das beim Troubleshooting.

```
iSCSI-Server (iscsi.example.com):
eth0: 172.17.1.1/24
eth1: 172.17.2.1/24

node1 (station5.example.com):
eth0: 192.168.0.30/24
eth1: 172.17.1.10/24
eth2: 172.17.2.10/24

node2 (station6.example.com):
eth0: 192.168.0.40/24
eth1: 172.17.1.20/24
eth2: 172.17.2.20/24

vm1 (station50.example.com):
eth0: 192.168.0.50

vm2 (station51.example.com):
eth0: 192.168.0.51

Webserver (www.example.com):
Floating IP: 192.168.0.60
```

Listing 3 zeigt die fertige Konfigurationsdatei des DomU-Cluster. Über das Tool Conga lässt sich nun anzeigen, auf welcher Maschine der Dienst aktiv ist, und über das Auswahlmenü innerhalb der Service-Konfiguration ist es möglich, den Dienst auch von einer virtuellen Maschine zur anderen zu migrieren.

Auf einer Gastmaschine ist dem Admin das Shell-Tool »clustat« behilflich, wenn es darum geht, eine Übersicht über die laufenden Services zu bekommen. Mithilfe von »cluvcadm« wiederum kann man den Service testweise auf eine andere Maschine des Cluster migrieren lassen. Mithilfe des Fencing Agent »fence_xvm« schließlich ist es möglich, von der Shell aus zu überprüfen, ob das Fence-System der Xen-Gäste funktioniert.

Fazit

Mit einer Kombination aus der Virtualisierungslösung Xen und der neuen Red Hat Cluster Suite lassen sich wie gezeigt recht leicht und schnell auch anspruchsvolle Cluster-Implementierungen aufsetzen. Selbst wer nur einmal zu Demozwecken oder für eine Schulung Cluster aufsetzen möchte, ist mit der in diesem Beitrag vorgestellten Lösung gut und zudem kostengünstig bedient. Weitere Informationen zu den dabei eingesetzten Technologien finden sich unter **(5)**. *(jcb)*

Infos

(1) iSCSI-target-Software: (http://iscsitarget.sourceforge.net) (http://people.redhat.com/tscherf/packages/rhel/iscsi/)

(2) Conga: (http://sources.redhat.com/cluster/conga/)

(3) Conga User Manual: (http://sources.redhat.com/cluster/conga/doc/user_manual.html)

(4) Virtualization Guide: (http://www.redhat.com/docs/manuals/enterprise/RHEL-5-manual/Virtualization-en-US/index.html)

(5) Red-Hat-Cluster: (http://sources.redhat.com/cluster/)

(6) system-config-cluster Manual: (http://www.redhat.com/docs/manuals/enterprise/RHEL-5-manual/Cluster_Administration/ch-config-scc-CA.html)

(7) Xen Setup Guide: (http://www.redhat.com/docs/manuals/enterprise/RHEL-5-manual/Virtualization-en-US/index.html)